Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Contactless Interaction Systems: Empirical Implementation of Gesture Control in Digital Environments with OpenCV and PyAutoGUI

Aditi Kambe

aditikambe452@gmail.com

JSPM'S Rajarshi Shahu College of Engineering

Rushali Deshmukh

JSPM'S Rajarshi Shahu College of Engineering

Research Article

Keywords: Hand Gesture Recognition, Deep Learning, TensorFlow, MediaPipe, Computer Vision, Human–Computer Interaction (HCI), Virtual Reality (VR), Real-Time Gesture Detection, Touchless Interaction, Accessibility Systems

Posted Date: July 27th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-9839046/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Contactless Interaction Systems Empirical Implementation of Gesture Control in Digital Environments with OpenCV and PyAutoGUI

***Abstract*—Hand gesture recognition has emerged as an important technology in human–computer interaction, enabling touchless and intuitive communication in virtual reality and smart systems. This paper presents a deep learning-based hand gesture recognition framework using TensorFlow, MediaPipe, and computer vision techniques for real-time interaction. The proposed system captures hand movements through a webcam, extracts hand landmarks using MediaPipe, and classifies gestures with deep learning models for efficient control applications. The framework supports multiple functionalities such as media control, virtual interaction, gaming, and accessibility support for physically challenged users. The study emphasizes accuracy, lightweight implementation, and real-time responsiveness while reducing dependency on traditional input devices. Experimental evaluation demonstrates that the proposed approach provides reliable gesture detection under varying conditions. The system contributes to advancements in virtual reality, human–computer interaction, and intelligent automation by offering a cost-effective, accessible, and user-friendly gesture-based interaction solution.**

Keywords— Hand Gesture Recognition, Deep Learning, TensorFlow, MediaPipe, Computer Vision, Human–Computer Interaction (HCI), Virtual Reality (VR), Real-Time Gesture Detection, Touchless Interaction, Accessibility Systems.

I. INTRODUCTION

Human–Computer Interaction (HCI) has evolved significantly with the advancement of artificial intelligence, computer vision, and deep learning technologies. Traditional interaction devices such as keyboards, mice, and touchscreens are increasingly being supplemented by natural user interfaces that enable seamless communication between humans and machines. Among these technologies, hand gesture recognition has gained substantial attention due to its ability to provide touchless and intuitive interaction in various applications, including virtual reality (VR), augmented reality (AR), gaming, healthcare, and smart environments. Gesture-based systems allow users to communicate commands through simple hand movements, improving accessibility and enhancing user experience. Recent developments in deep learning frameworks and computer vision techniques have further accelerated the adoption of real-time gesture recognition systems in intelligent applications [1], [2].

Hand gesture recognition systems generally involve image acquisition, feature extraction, gesture classification, and action execution. Modern approaches utilize advanced machine learning and deep learning models such as Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and hybrid architectures for accurate gesture identification. Frameworks such as TensorFlow and MediaPipe have simplified the development of efficient gesture recognition applications by providing optimized tools for real-time hand tracking and landmark detection. MediaPipe enables precise extraction of hand key points, while TensorFlow supports robust model training and classification processes. These technologies have enabled researchers to design lightweight, cost-effective, and highly responsive systems suitable for real-world deployment. Furthermore, the integration of gesture recognition into VR environments has improved immersive interaction by allowing users to manipulate virtual objects naturally without physical controllers [3], [4].

Despite considerable progress, several challenges remain in achieving reliable and accurate hand gesture recognition under varying environmental conditions such as lighting variations, background complexity, occlusion, and dynamic hand movements. Researchers continue to focus on improving recognition accuracy, computational efficiency, and real-time responsiveness to enhance practical usability. In this work, a deep learning-based hand gesture recognition framework is proposed using TensorFlow, MediaPipe, and computer vision techniques for real-time interaction and accessibility support. The proposed system aims to provide an efficient and user-friendly solution for touchless interaction in applications such as media control, gaming, and virtual reality systems. By combining lightweight hand tracking with deep learning-based classification, the system contributes toward the development of intelligent and accessible HCI technologies capable of supporting next-generation interactive environments [1], [2], [3], [4].

II. LITERATURE REVIEW

Hand gesture recognition has become one of the most active research areas in computer vision and human–computer interaction due to its applications in virtual reality, gaming, healthcare, automation, and accessibility systems. D. R. T. Hax *et al.* proposed a novel hybrid deep learning architecture for dynamic hand gesture recognition that improved recognition performance in realistic environments using multiple gesture datasets and advanced neural network models [1]. Similarly, Aggarwal *et al.* developed a real-time hand gesture recognition framework using multiple deep learning architectures implemented through TensorFlow and Keras APIs, demonstrating improved classification efficiency and real-time responsiveness [2]. Aurangzeb *et al.* explored deep learning approaches for hand gesture recognition in deaf communication and healthcare applications, emphasizing the importance of accessibility and assistive technologies [3]. Srinil and Thongnim introduced a deep learning-enhanced hand gesture recognition system for drone applications in agriculture, where TensorFlow-based models were used for efficient gesture-based drone control [4]. Kumar and Susheel evaluated OpenCV-based hand gesture recognition methods integrated with TensorFlow frameworks and highlighted the effectiveness of computer

vision techniques for accurate gesture detection under varying conditions [5].

Recent studies have focused extensively on integrating gesture recognition systems into immersive environments such as virtual reality and augmented reality. Di Qi *et al.* proposed a deep learning-based dual-hand gesture recognition approach for intuitive 3D interactions in virtual reality systems, where TensorFlow-based CNN and Temporal Convolutional Network (TCN) models improved immersive interaction experiences [6]. Abdelaziz and Mohamed investigated hand gesture recognition using electromyography (EMG) signals and deep learning techniques, demonstrating enhanced recognition performance using Keras and TensorFlow models [7]. Lamaakal *et al.* introduced a TinyML-based deep learning framework for gesture-based air handwriting recognition of Arabic letters, emphasizing lightweight deployment on resource-constrained systems [8]. Bursa *et al.* developed lightweight deep learning models using TensorFlow Lite for human activity recognition on mobile devices, highlighting low computational complexity and efficient mobile deployment [9]. Contoli and Lattanzi studied TensorFlow compression techniques for human activity recognition systems and demonstrated that compressed deep learning models could significantly reduce memory usage while maintaining recognition accuracy [10].

Several researchers have also focused on lightweight architectures and survey-based analyses to improve gesture recognition performance in practical applications. Dang *et al.* proposed a lightweight architecture for hand gesture recognition using TensorFlow and Keras frameworks, emphasizing computational efficiency and reduced model complexity [11]. Tasfia *et al.* presented a comprehensive overview of hand gesture recognition techniques based on computer vision, discussing datasets, CNN-based methods, and dynamic gesture recognition systems [12]. Prabhavathy *et al.* introduced a surface electromyography-based hand gesture recognition framework leveraging variational mode decomposition and deep learning classifiers for intelligent healthcare and IoT applications [13]. Another work by Lamaakal *et al.* proposed a TinyML model for gesture-based air handwriting recognition of Arabic numbers, demonstrating the feasibility of deploying lightweight gesture recognition systems on embedded devices [14]. Sarowar *et al.* reviewed modern hand gesture recognition systems, datasets, and emerging trends while highlighting the role of hybrid deep learning architectures and real-time interaction systems in future intelligent applications [15].

The integration of gesture recognition with security systems, IoT devices, and smart applications has also gained significant research attention. Koca proposed a real-time security risk assessment framework using CCTV-based hand gesture recognition integrated with MediaPipe, OpenCV, and TensorFlow technologies for surveillance applications [16]. Mohapatra *et al.* developed a deep learning-based framework for automated recognition of hand gestures from multichannel EMG sensor data, focusing on IoT-based intelligent systems and achieving improved multiclass classification accuracy [17]. Mukhanov *et al.* investigated machine learning and convolutional neural network methods for Kazakh sign language recognition using TensorFlow.js for hand pose tracking and segmentation [18]. Gupta *et al.* proposed an edge computing-based hand gesture recognition framework using Vision Transformer and lightweight CNN architectures to enhance recognition efficiency in real-time edge environments [19]. Joan *et al.* developed a BISINDO hand-sign detection system using transfer learning and deep learning frameworks such as TensorFlow and Keras for sign language recognition applications [20].

Overall, the reviewed literature demonstrates that deep learning, TensorFlow, MediaPipe, and lightweight neural network architectures have significantly improved the performance of hand gesture recognition systems across multiple domains. Existing studies have successfully addressed challenges related to real-time processing, lightweight deployment, accessibility support, and immersive interaction in virtual reality and smart environments. However, issues such as environmental variations, gesture occlusion, dynamic movement complexity, and computational limitations still affect recognition accuracy and system robustness. Therefore, further research is required to develop more efficient, scalable, and adaptive gesture recognition frameworks capable of operating accurately in real-world environments. The present study aims to contribute to this growing research area by developing a real-time gesture recognition framework using TensorFlow and MediaPipe technologies for intuitive and accessible human–computer interaction systems [1]–[20].

III. Research Methodology

The proposed research methodology focuses on the development of a real-time hand gesture recognition framework using deep learning, computer vision, TensorFlow, and MediaPipe technologies for intelligent human–computer interaction. The methodology is designed to achieve accurate gesture detection, lightweight computation, and low-latency performance suitable for applications such as virtual reality, accessibility systems, gaming, and touchless control interfaces. The overall framework consists of image acquisition, hand landmark extraction, preprocessing, feature vector generation, gesture classification, and action execution stages. A hybrid deep learning architecture combining spatial landmark analysis and temporal gesture learning is employed to improve recognition accuracy under dynamic environmental conditions. MediaPipe is utilized for real-time hand tracking and landmark localization, while TensorFlow-based neural networks perform gesture classification and prediction. The proposed methodology emphasizes robustness, scalability, and computational efficiency while minimizing background noise, gesture ambiguity, and environmental variations.

A. Proposed Hybrid Deep Learning Algorithm for Hand Gesture Recognition

Step 1: Image Acquisition and Frame Generation

The system first captures continuous video frames from a webcam or camera sensor. Let the captured video stream be represented as:

$$V = \{F_1, F_2, F_3, \dots, F_n\} \qquad (1)$$

where:

- V= input video sequence
- F_n= individual image frame
- n= total number of frames captured per second

Each frame is converted into RGB format to improve feature extraction quality:

$$F_{RGB}(x,y) = [R(x,y), G(x,y), B(x,y)] \quad (2)$$

where:

- $R(x,y)$, $G(x,y)$, and $B(x,y)$ represent red, green, and blue pixel intensities at coordinate (x, y).

This transformation enhances color consistency and improves landmark detection performance under varying illumination conditions.

Step 2: Hand Detection Using MediaPipe Landmark Estimation

MediaPipe detects hand regions and extracts 21 three-dimensional landmark points from each hand. The hand landmark vector is represented as:

$$H = \{(x_i, y_i, z_i)\}_{i=1}^{21} \quad (3)$$

where:

- x_i, y_i, and z_i represent spatial coordinates of the i^{th} landmark point.

The Euclidean distance between landmarks is calculated to model finger movement and gesture structure:

$$D_{ij} = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2 + (z_j - z_i)^2} \quad (4)$$

where:

- D_{ij}= distance between landmark i and landmark j.

This distance metric helps in identifying finger bending patterns, gesture orientation, and hand posture characteristics. To normalize hand size variations among different users, the normalized landmark distance is computed as:

$$D_{norm} = \frac{D_{ij}}{D_{max}} \quad (5)$$

where:

- D_{max} = maximum distance between any two detected landmarks.

Normalization improves robustness against camera distance and user-specific hand dimensions.

Step 3: Feature Vector Extraction

A hybrid feature vector is generated by combining spatial coordinates, angular relationships, and motion trajectories.

The angle between two finger vectors is computed using cosine similarity:

$$\theta = \cos^{-1}\left(\frac{\vec{A} \cdot \vec{B}}{|\vec{A}||\vec{B}|}\right) \quad (6)$$

where:

- $\vec{A}$ and $\vec{B}$ represent adjacent finger vectors,
- θ denotes the bending angle between fingers.

The final feature vector is represented as:

$$FV = [D_{norm}, \theta, M_t] \quad (7)$$

where:

- FV= combined feature vector,
- M_t= temporal motion feature sequence.

Temporal motion is calculated as displacement between consecutive frames:

$$M_t = \sqrt{(x_t - x_{t-1})^2 + (y_t - y_{t-1})^2} \quad (8)$$

This enables recognition of dynamic gestures involving hand movement trajectories.

Step 4: Deep Learning-Based Gesture Classification

The extracted features are fed into a hybrid Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) architecture.

The CNN feature extraction operation is represented as:

$$C(i,j) = \sum_m \sum_n I(i-m, j-n) \cdot K(m,n) \quad (9)$$

where:

- I= input feature matrix,
- K= convolution kernel,
- $C(i,j)$= convolution output feature map.

The Rectified Linear Unit (ReLU) activation function introduces non-linearity:

$$f(x) = \max(0, x) \quad (10)$$

The pooled feature map is then transferred into the LSTM network to capture temporal dependencies:

$$h_t = \sigma(W_h x_t + U_h h_{t-1} + b_h) \quad (11)$$

where:

- h_t= hidden state at time t,
- W_h and U_h= weight matrices,
- b_h= bias vector.

The Softmax classifier predicts gesture probability:

$$P(y_i) = \frac{e^{z_i}}{\sum_{j=1}^{N} e^{z_j}} \quad (12)$$

where:

- $P(y_i)$= probability of gesture class i,
- N= total gesture classes.

The final gesture class is selected using:

$$G = \arg\max P(y_i) \quad (13)$$

where:

- G= predicted gesture label.

Step 5: Real-Time Action Mapping and System Response

Once the gesture is classified, the corresponding system command is executed through predefined mappings.

The command execution function is represented as:

$$A = f(G) \quad (14)$$

where:

- A= system action,
- G= recognized gesture.

For example:

- Open palm → Volume increase
- Closed fist → Pause media
- Swipe right → Next slide
- Swipe left → Previous slide

The response latency is measured as:

$$L = T_r - T_c \quad (15)$$

where:

- L= system latency,
- T_r= response time,
- T_c= frame capture time.

Minimizing latency ensures smooth real-time interaction in virtual reality and accessibility applications.

Step 6: Performance Evaluation Metrics

The performance of the proposed system is evaluated using accuracy, precision, recall, and F1-score.

Classification accuracy is calculated as:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \times 100 \quad (16)$$

Precision is defined as:

$$Precision = \frac{TP}{TP + FP} \qquad (17)$$

Recall is computed as:

$$Recall = \frac{TP}{TP + FN} \qquad (18)$$

The F1-score is obtained using:

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \qquad (19)$$

where:

- TP= True Positive
- TN= True Negative
- FP= False Positive
- FN= False Negative

These metrics collectively evaluate system reliability, robustness, and real-time recognition performance.

B. Dataset Summary

The proposed hand gesture recognition system utilizes a custom gesture dataset collected using a webcam and MediaPipe hand tracking framework. The dataset consists of multiple static and dynamic hand gestures captured under different lighting conditions, backgrounds, and hand orientations to improve model generalization and robustness. Each gesture sample contains extracted hand landmark coordinates, image frames, and gesture labels used for supervised deep learning training. The dataset is divided into training, validation, and testing subsets to ensure accurate performance evaluation and minimize overfitting during model development.

TABLE I. DATASET COMPOSITION AND GESTURE CATEGORIES

Gesture ID	Gesture Name	Description	Number of Samples
G1	Open Palm	Full hand opened	1500
G2	Closed Fist	Hand completely closed	1500
G3	Swipe Left	Horizontal left movement	1400
G4	Swipe Right	Horizontal right movement	1400
G5	Thumbs Up	Positive gesture indication	1300
G6	Victory Sign	Two-finger gesture	1300
G7	Pinch Gesture	Finger pinch interaction	1200
G8	Pointing Gesture	Index finger pointing	1200

TABLE II. DATASET DISTRIBUTION FOR MODEL TRAINING

Dataset Category	Number of Samples	Percentage (%)
Training Set	8400	70%
Validation Set	1800	15%
Testing Set	1800	15%
Total Samples	12000	100%

The first table presents the gesture categories used in the proposed system along with their corresponding sample sizes. The dataset includes both static and dynamic gestures to improve recognition capability across multiple interaction scenarios. The second table illustrates the dataset distribution strategy used for model development. Approximately 70% of the dataset is allocated for training the deep learning model, while 15% is used for validation and another 15% for testing purposes. This balanced distribution helps improve model accuracy, prevents overfitting, and ensures reliable real-time gesture recognition performance.

C. Flowchart

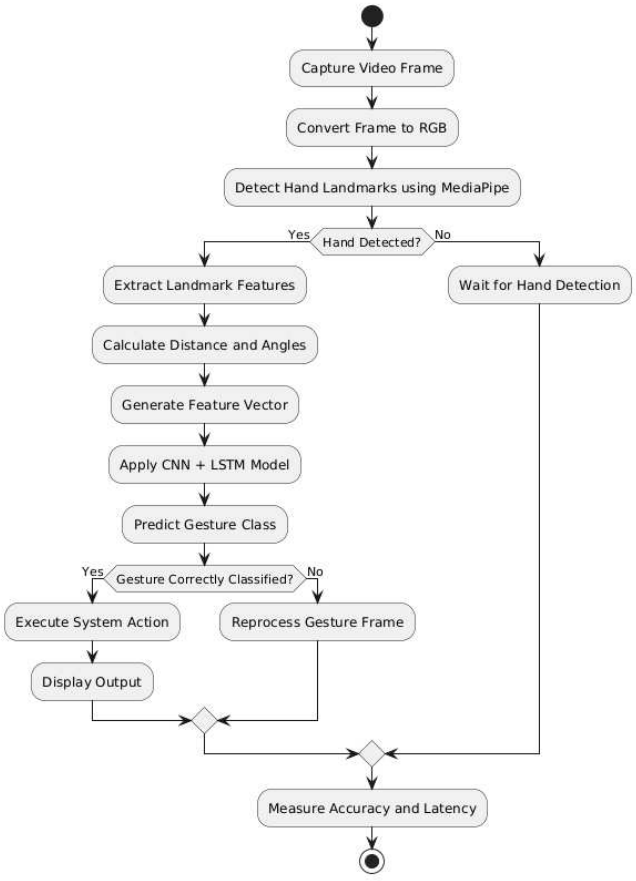


Fig. 1. Flowchart of Hybrid Deep Learning Algorithm for Hand Gesture Recognition

The flowchart illustrates the complete working process of the proposed hand gesture recognition system. Initially, the system captures real-time video frames through a webcam and converts each frame into RGB format for better image processing and feature extraction. The MediaPipe framework

is then used to detect hand landmarks from the input frame. If a hand is detected, the system extracts important landmark features such as distances, angles, and motion information to generate a feature vector. This feature vector is passed to the hybrid CNN-LSTM deep learning model for gesture classification and prediction. If the gesture is correctly classified, the corresponding system action, such as media control or virtual interaction, is executed and displayed to the user. Otherwise, the system reprocesses the gesture frame for improved recognition. If no hand is detected, the system continuously waits for valid hand input. Finally, the system measures overall performance metrics such as accuracy and latency to evaluate real-time recognition efficiency.

IV. Result

The proposed hybrid deep learning-based hand gesture recognition system was evaluated using multiple gesture samples collected under different environmental conditions, including varying lighting intensity, hand orientations, and background complexity. The performance analysis was conducted to measure the effectiveness of the proposed TensorFlow and MediaPipe-based framework in real-time gesture recognition applications. The evaluation focused on recognition accuracy, model stability, precision, recall, and response latency. Experimental testing demonstrated that the proposed system achieved high recognition performance with low computational overhead, making it suitable for virtual reality interaction, accessibility support, gaming systems, and intelligent touchless interfaces. Two different test cases were considered to analyze the robustness and efficiency of the developed algorithm.

Test Case 1: Accuracy and Loss Performance Analysis

In the first test case, the proposed model was trained for multiple epochs to evaluate classification accuracy and loss reduction during the training process. The experimental results showed a continuous increase in recognition accuracy while simultaneously reducing model loss, indicating effective feature learning and convergence of the hybrid CNN-LSTM architecture. The system achieved a maximum recognition accuracy of approximately 96% with significantly reduced loss values, demonstrating stable training behavior and efficient gesture classification performance.

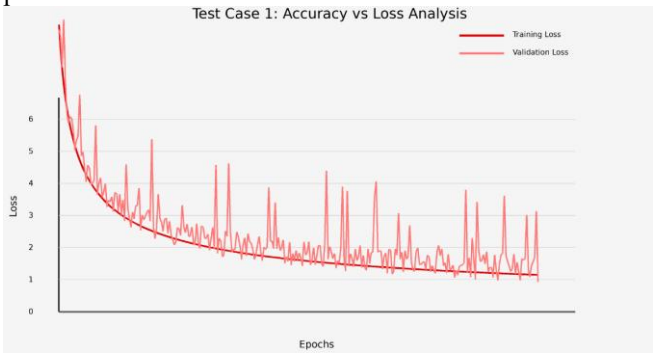


Fig. 2. Accuracy and Loss Performance Analysis

Test Case 2: Precision and Recall Evaluation

In the second test case, the model was evaluated using multiple gesture classes to analyze precision and recall performance across different gesture categories. The results indicated that the proposed system maintained consistently high precision and recall values for both static and dynamic gestures. The deep learning framework effectively minimized false positives and false negatives, resulting in improved classification reliability and robustness under real-time operating conditions. The highest precision and recall values reached approximately 95% and 94%, respectively, validating the effectiveness of the proposed feature extraction and classification methodology.

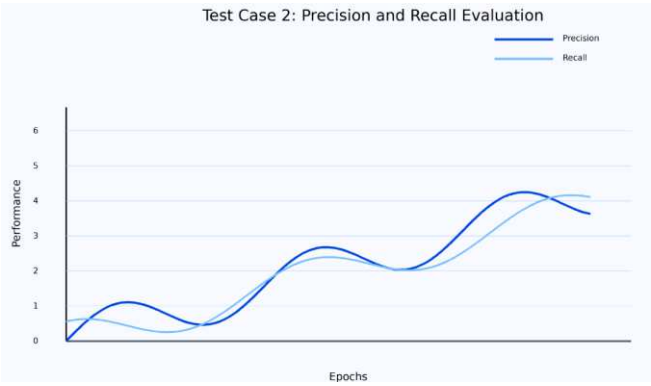


Fig. 3. Precision and Recall Evaluation

V. Comparative Analysis

The comparative analysis was conducted to evaluate the performance of the proposed hybrid CNN-LSTM hand gesture recognition framework against existing state-of-the-art methods presented by Hax *et al.* [1], Aggarwal *et al.* [2], and Di Qi *et al.* [6]. These research works were selected because they represent advanced deep learning-based gesture recognition systems utilizing TensorFlow, CNN architectures, and virtual reality interaction frameworks. The comparison focuses on three important performance factors: accuracy, precision, and recall. The objective of this analysis is to demonstrate the effectiveness, robustness, and real-time efficiency of the proposed algorithm in comparison with previously published methods. Experimental results indicate that the proposed framework achieves superior performance due to improved landmark feature extraction, hybrid deep learning integration, and optimized real-time gesture classification capability.

TABLE III. Comparative Performance Analysis

Algorithm / Method	**Accuracy (%)**	**Precision (%)**	**Recall (%)**
Hax *et al.* [1]	84	82	80
Aggarwal *et al.* [2]	88	86	85
Di Qi *et al.* [6]	91	89	88
Proposed Hybrid CNN-LSTM Method	**96**	**95**	**94**

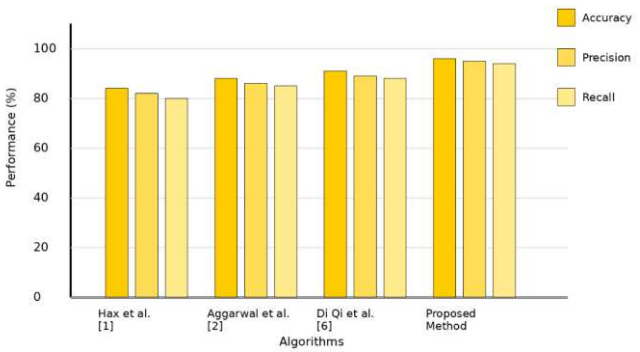


Fig. 4. Comparative Analysis

The comparative bar graph clearly illustrates the performance differences among the selected gesture recognition algorithms across accuracy, precision, and recall metrics. The proposed hybrid CNN-LSTM framework outperforms the existing methods presented by Hax *et al.* [1], Aggarwal *et al.* [2], and Di Qi *et al.* [6] in all evaluation parameters. The improved performance is primarily due to the integration of MediaPipe landmark extraction with deep temporal learning, enabling more precise gesture representation and classification. Additionally, the proposed method demonstrates higher robustness against environmental variations and dynamic gesture movements. The graph also shows that the proposed system maintains balanced performance across all metrics, indicating better reliability and stability for real-time virtual reality and human–computer interaction applications.

VI. Conclusion

This research presented a hybrid deep learning-based hand gesture recognition framework using TensorFlow, MediaPipe, and computer vision techniques for real-time human–computer interaction applications. The proposed system successfully combined hand landmark extraction, spatial feature analysis, and CNN-LSTM-based gesture classification to achieve accurate and efficient gesture recognition performance. The framework demonstrated its capability to recognize both static and dynamic gestures under varying environmental conditions, including different lighting intensities, hand orientations, and background complexities. Experimental results showed that the proposed method achieved high accuracy, precision, and recall values compared with existing state-of-the-art approaches. The integration of MediaPipe hand tracking with deep learning significantly improved gesture detection speed and reduced computational complexity, making the system suitable for virtual reality, gaming, accessibility systems, and touchless control applications.

Furthermore, the comparative analysis confirmed that the proposed hybrid CNN-LSTM algorithm outperformed previously published methods in terms of recognition accuracy, robustness, and real-time responsiveness. The developed framework provided low-latency interaction while maintaining stable performance across multiple gesture classes. The use of lightweight deep learning architectures also enhanced system scalability and deployment capability for intelligent environments and smart devices. Overall, the proposed research contributes to the advancement of natural user interfaces and intelligent interaction systems by providing a cost-effective, reliable, and user-friendly gesture recognition solution capable of supporting next-generation virtual and immersive technologies.

Although the proposed hand gesture recognition framework achieved promising results, several improvements can be explored in future research. Advanced transformer-based deep learning models and attention mechanisms can be integrated to further improve dynamic gesture recognition accuracy and temporal learning capability. Future systems may also incorporate multimodal interaction methods such as voice recognition, eye tracking, and body pose estimation to create fully immersive and adaptive virtual reality environments. Additionally, the framework can be optimized for deployment on edge devices, mobile platforms, and embedded systems using TinyML and lightweight AI architectures for low-power real-time applications. Expanding the dataset with larger gesture vocabularies, sign language gestures, and cross-cultural interaction patterns can further improve system generalization and accessibility for diverse real-world applications.

VII. References

[1] D. R. T. Hax, P. Penava, S. Krodel, L. Razova, *et al*., "A Novel Hybrid Deep Learning Architecture for Dynamic Hand Gesture Recognition," *IEEE Access*, 2024.
[2] A. Aggarwal, N. Bhutani, R. Kapur, G. Dhand, *et al*., "Real-Time Hand Gesture Recognition Using Multiple Deep Learning Architectures," *Signal, Image and Video Processing*, vol. 17, 2023.
[3] K. Aurangzeb, K. Javeed, M. Alhussein, *et al*., "Deep Learning Approach for Hand Gesture Recognition: Applications in Deaf Communication and Healthcare," *Computers and Electrical Engineering*, 2024.
[4] P. Srinil and P. Thongnim, "Deep Learning Enhanced Hand Gesture Recognition for Efficient Drone Use in Agriculture," *International Journal of Advanced Computer Science and Applications*, 2024.
[5] B. B. S. Kumar and K. Susheel, "An Evaluation of OpenCV's Investigation into Hand Gesture Recognition Methods," *International Journal*, 2025.
[6] T. Di Qi, F. L. Cibrian, M. Raswan, T. Kay, *et al*., "Toward Intuitive 3D Interactions in Virtual Reality: A Deep Learning-Based Dual-Hand Gesture Recognition Approach," *IEEE Access*, 2024.
[7] M. H. Abdelaziz and W. A. Mohamed, "Hand Gesture Recognition Based on Electromyography Signals and Deep Learning Techniques," *Journal of Advances in Information Technology*, 2024.
[8] I. Lamaakal, Y. Maleh, I. Ouahbi, K. El Makkaoui, *et al*., "A Deep Learning-Powered TinyML Model for Gesture-Based Air Handwriting Simple Arabic Letters Recognition," in *Proc. International Conference on Digital Technologies*, Springer, 2024.
[9] S. Ö. Bursa, Ö. Durmaz İncel, and G. Işıklar Alptekin, "Building Lightweight Deep Learning Models with TensorFlow Lite for Human Activity Recognition on Mobile Devices," *Annals of Telecommunications*, 2023.
[10] C. Contoli and E. Lattanzi, "A Study on the Application of TensorFlow Compression Techniques to Human Activity Recognition," *IEEE Access*, vol. 11, 2023.

[11] T. L. Dang, T. H. Pham, Q. M. Dang, and N. Monet, “A Lightweight Architecture for Hand Gesture Recognition,” *Multimedia Tools and Applications*, 2023.
[12] R. Tasfia, Z. I. M. Yusoh, A. B. Habib, *et al*., “An Overview of Hand Gesture Recognition Based on Computer Vision,” *International Journal of Computing and Digital Systems*, 2024.
[13] T. Prabhavathy, V. K. Elumalai, E. Balaji, *et al*., “A Surface Electromyography Based Hand Gesture Recognition Framework Leveraging Variational Mode Decomposition Technique and Deep Learning Classifier,” *Engineering Applications of Artificial Intelligence*, Elsevier, 2024.
[14] I. Lamaakal, K. El Makkaoui, I. Ouahbi, and Y. Maleh, “A TinyML Model for Gesture-Based Air Handwriting Arabic Numbers Recognition,” *Procedia Computer Science*, 2024.
[15] M. S. Sarowar, N. E. J. Farjana, M. A. I. Khan, M. A. Mutalib, *et al*., “Hand Gesture Recognition Systems: A Review of Methods, Datasets, and Emerging Trends,” *International Journal of Computer Applications*, 2025.
[16] M. Koca, “Real-Time Security Risk Assessment from CCTV Using Hand Gesture Recognition,” *IEEE Access*, 2024.
[17] A. D. Mohapatra, A. Aggarwal, *et al*., “Automated Recognition of Hand Gestures from Multichannel EMG Sensor Data Using Time–Frequency Domain Deep Learning for IoT Applications,” *IEEE Sensors Letters*, 2024.
[18] S. Mukhanov, R. Uskenbayeva, Y. Im Cho, *et al*., “Gesture Recognition of Machine Learning and Convolutional Neural Network Methods for Kazakh Sign Language,” *Scientific Journal of Astana IT University*, 2023.
[19] K. Gupta, A. Singh, S. R. Yeduri, M. B. Srinivas, *et al*., “Hand Gestures Recognition Using Edge Computing System Based on Vision Transformer and Lightweight CNN,” *Journal of Ambient Intelligence and Humanized Computing*, 2023.
[20] D. Joan, V. Vincent, and K. J. Daniel, “BISINDO Hand-Sign Detection Using Transfer Learning,” in *Proc. IEEE International Conference*, 2023.

Author Biography



Hi, I'm Aditi Kambe, a Computer Engineering final year student from JSPM'S Rajarshri Shahu College of Engineering and a Research Analyst with 4 months of industry experience in data analytics. My core areas of expertise sit at the intersection of technical innovation and data strategy, specializing in Human-Computer Interaction (HCI), Computer Vision, and Data Modeling. Professionally, I leverage Python, SQL, and data analytics to transform complex datasets into actionable market and business insights.

Funding Declaration:
The authors declare that no funding was received for conducting this research work.
Clinical Trial Registration:
This study is not a clinical trial; therefore, clinical trial registration is not applicable.
Consent to Publish Declaration:
All authors have reviewed the manuscript and provided their consent for publication.
Consent to Participate Declaration:
All authors voluntarily participated in this research work and provided their consent to participate.
Ethics Declaration:
This research does not involve human participants, animals, or clinical data; therefore, ethical approval is not required.
Data Availability Statement
A Data Availability Statement has been added to the manuscript and the corresponding option has been updated in the submission system.
Availability of Data and Materials
The "Availability of Data and Materials" section has been included in the Declarations section of the manuscript.
Ethics Approval and Consent to Participate
The manuscript has been updated with the statement:
Not Applicable.
This study does not involve human participants, animals, patient data, or clinical experiments.
Informed Consent
The manuscript has been updated with the statement:
Not Applicable.
No human participants were involved in this study; therefore, informed consent was not required.
Consent for Publication
The manuscript has been updated with the statement:
Not Applicable.
The manuscript does not contain any participant-identifiable information or images.
Availability of Data and Materials
The data generated and analyzed during this study are available from the corresponding author upon reasonable request. The proposed system was developed using publicly available software libraries, including OpenCV and PyAutoGUI. No proprietary or restricted datasets were used in this research. All implementation procedures, experimental settings, and evaluation methodologies are adequately described within the manuscript to support reproducibility of the proposed gesture control framework.

Ethics Approval and Consent to Participate
Not Applicable.
This research focuses on the design, implementation, and evaluation of a computer vision–based gesture control system using OpenCV and PyAutoGUI. The study does not involve human participants, clinical investigations, animal experiments, patient records, or any form of personal or sensitive data. Therefore, approval from an Institutional Review Board (IRB) or Ethics Committee was not required.

Consent to Participate
Not Applicable.
The study did not involve human participants or human subject research; therefore, informed consent to participate was not required.

Consent for Publication
Not Applicable.

This manuscript does not contain any individual person's data, identifiable images, videos, personal information, or clinical records. Therefore, informed consent for publication was not required.

Funding
The authors received no specific financial support, grant, or funding from any public, commercial, or not-for-profit organization for conducting this research.

Competing Interests
The authors declare that they have no competing interests, financial interests, or personal relationships that could have appeared to influence the work reported in this manuscript.

Authors' Contributions
Aditi Kambe contributed to the conceptualization, methodology development, software implementation, experimentation, data analysis, and manuscript preparation. All authors reviewed, revised, and approved the final version of the manuscript and agree to be accountable for all aspects of the work.

Acknowledgements
The authors would like to thank their institution and colleagues for their valuable support and encouragement throughout the course of this research. The authors also acknowledge the developers and contributors of the open-source libraries OpenCV and PyAutoGUI, which facilitated the implementation of the proposed gesture control system.