



Head Gesture Recognition with Dense Net Ensemble Classifier from IMU Sensor Data

Lilly Mondithoka¹, Ch pavani²

¹M. Tech Student, ²Assistant Professor, ^{1,2} Department of Electronics and communication Engineering, Mother Teresa Institute of Science and Technology, Sathupally, Khammam 507303, Telangana, India.

ABSTRACT

Head gesture classification using Inertial Measurement Unit (IMU) sensor data has become increasingly important for touchless human-machine interaction, especially in wearable and assistive technologies. Recent studies show that nearly 65% of gesture-based interfaces still rely on vision-based systems, which are less reliable in low-light or occluded environments, while sensor-based interfaces improve accuracy by up to 20% in such conditions. This study introduces a novel framework for accurate head gesture classification using IMU data, focusing on robust preprocessing, effective data balancing, and advanced model design. Initially, the raw IMU signals are denoised and standardized through a dedicated preprocessing pipeline to ensure consistency across all gesture categories. To address the class imbalance commonly observed in real-world gesture datasets, the Synthetic Minority Oversampling Technique (SMOTE) is applied, which synthetically augments underrepresented gesture classes such as Jump Next Left and Stop Video, ensuring more balanced and representative training samples. The baseline models such as Perceptron and MLP are evaluated, revealing limitations in distinguishing subtle variations in gestures like MoveLeft_2s and MoveRight_2s, due to their shallow architecture and low expressive power. To overcome this, the proposed DenseNet-RFC framework is introduced, where DenseNet extracts deep spatial-temporal features from the IMU signal segments, and the Random Forest Classifier (RFC) robustly classifies these features across eight gesture classes:

MoveRight_2s, MoveLeft_2s, Jump Next Right, Jump Next Left, Rise Volume-up, Rise Volume-down, Start Video, and Stop Video. This fusion of deep representation with ensemble classification leads to significantly improved accuracy, generalization, and interpretability of gesture recognition in noisy and imbalanced IMU data environments.

1. Introduction

Advances in gesture recognition have transformed human-machine interaction, with the global gesture recognition and touchless sensing market reaching approximately \$23.55 billion in 2020, and projections estimating growth to \$89.3 billion by 2031, with sensor-based systems accounting for 42 % of the market share. Among diverse technologies, Inertial Measurement Units (IMUs)—comprising accelerometers and gyroscopes—have become especially prevalent in handheld and wearable devices such as smartphones, VR headsets, and game controllers, thanks to their compact form, low cost, and immunity to lighting variability.

IMU-driven head gesture recognition demonstrates impressive performance across multiple applications. In some studies, recognition rates exceeded 90 % for multiple classifiers both in person-dependent and person-independent setups. Other works achieved remarkable results; for instance, DTW-based methods delivered 100 % accuracy across six head movement types. Assistive systems—from behavior-assistive robots to smart wheelchairs—have also employed IMUs for head motion control, enabling intuitive interaction for users with mobility impairments.



Nevertheless, challenges remain. IMU-only systems often face issues with drift and positional errors, particularly when integrated or double-integrated for tracking, necessitating sophisticated sensor fusion or correction mechanism. In addition, classification accuracy and robustness highly depend on the feature selection strategy richer statistical features such as frequency-domain metrics yield better performance yet may increase computational demands and reduce real-time responsiveness.

2.LITERATURE SURVEY

Kadhim et al. [6] proposed a hybrid method that segmented dynamic gestures via an optimized Otsu threshold, extracted texture and contour features from RGB video, used Inception-v3 for feature extraction, and LSTM for temporal classification. The model recognised six gestures, reporting average precision 84.34% and 87.57% for impaired-finger gestures. It suffered from limited feature discrimination due to reliance on contour and texture only, leading to modest classification boundaries and lower separation among similar gestures.

Zhou et al. [7] developed a graph convolutional network (GCN) with attention mechanisms to classify head gestures from surface-EMG (sEMG) signals. The architecture modelled spatial connectivity among EMG channels and incorporated attention to weight salient activations for classification tasks. It incurred high computational complexity during feature selection and graph construction, slowing real-time performance and limiting suitability for on-device inference.

Biswas et al. [8] introduced a hybrid convolutional neural network (SP-CNN) with embedded attention modules to enhance human-robot collaboration. The architecture fused spatial convolution layers with attention weighting to extract discriminative features from sEMG for

gesture recognition in collaborative scenarios. It displayed instability in feature localisation, leading to inconsistent weighting of channels and misclassification when non-informative inputs dominated.

Kyranou et al. [9] provided a public EMG dataset capturing gestures with arm translation to evaluate EMG-based recognition methods. They offered rich multi-subject recordings across translational shifts, enabling evaluation of temporal and spatial variation in gesture signals. Lack of explicit feature extraction benchmarks led to grading inconsistency across methods, since feature saliency was not standardised and models lacked clear guidance on EMG electrode arrangement.

Li et al. [10] proposed cross-domain gesture recognition using WiFi signals and deep learning. The approach reconstructed Channel State Information using low-frequency components, filtered out high-frequency noise, and extracted reconstructed amplitude and phase features. A deep convolutional neural network learned gesture representations robust across domains. The model was evaluated across varying environments and reported high cross-domain accuracy. Feature extraction depended on low-frequency reconstruction only, leading to limited discriminative power when gestures produced subtle variations unaligned with retained components.

Prakash et al. [11] presented an optimized electrode configuration for wrist-wearable EMG-based gesture recognition using classical machine learning. They tested multiple electrode placements to extract spatial and temporal features like RMS, waveform length and frequency bands. Feature selection through sequential forward search identified the most informative features. A classifier (e.g. SVM or RF) achieved high accuracy on seen gestures in wearables. The sequential feature selection process increased computation overhead during training and



risked overfitting to specific electrode layouts, reducing generalisation.

Liu et al. [12] introduced a machine-learning integrated flexible sensor combining strain and pressure dual-mode sensing for head gesture assistance in unable-phonation scenarios. They captured concurrent strain and pressure signals, extracted multimodal features via handcrafted time-domain and frequency-domain metrics, then applied ML classifiers. Gesture classes related to phonation assistance showed high recall. System demonstrated synergy of dual-mode sensors. Feature fusion grading lacked systematic weighting, leading to inconsistent feature importance across participants and reduced robustness in classification.

Yang et al. [13] designed a spatio-temporal cross network with subject-aware contrastive (SAC) learning for sEMG-based recognition. They segmented sEMG sequences, applied rolling convolution reflecting circular electrode topology, then convolutional-recurrent blocks processed spatio-temporal patterns. SAC loss aligned feature embeddings across subjects. Evaluations showed robust inter-subject recognition performance. The sophisticated cross-subject contrastive mechanism increased training complexity and prolonged convergence, hindering scalability.

Dong et al. [14] proposed an AI-assisted flexible wearable mechanoluminescent strain sensor system that measured hand movements. The sensor converted strain into light emission signals; features derived from intensity and timing patterns fed into learning algorithms. They trained a lightweight classifier to map luminescent patterns to gestures, achieving real-time predictions. Feature extraction from luminescence signals lacked robustness under variable ambient lighting, leading to unstable classification boundaries.

Zhang et al. [15] introduced Wiopen, a Wi-Fi-based open-set gesture recognition framework. They extracted amplitude, phase, and Doppler shift features from CSI, used a deep network with open-set detection head to classify known gestures and detect unknown ones. The model learned domain-robust features for both classification and novelty detection. Evaluations showed strong open-set generalisation. Grading of open-set threshold selection remained manual and static, leading to trade-off issues between false reject and false accept rates under varying environmental conditions.

Gao et al. [16] proposed a multilayer bionic tunable strain sensor featuring mutually non-interfering conductive networks (CGGTM). They engineered distinct conductive layers to isolate signal channels, achieving ultra-low detection limit ($\sim 0.05\%$ strain) and high gauge factor ($>150\,000$). They coupled the sensor with machine learning classifiers to interpret gesture patterns from strain signals. They demonstrated accurate classification across a wide strain range (up to $\sim 364\%$), with real-time gesture recognition. Extraction of features across layers lacked robust cross-layer normalization, resulting in variability in feature scaling and degraded model stability.

Najib et al. [17] conducted a comprehensive review of sign language interpretation systems using machine learning and artificial intelligence. They categorized methods by modality—head gestures, facial expressions, lip reading and summarised both translation to speech/text and reverse processes. They evaluated real-world mobile implementations and highlighted integration gaps between research and deployed applications. They proposed a combined set of functionalities distilled from prior research directions. They did not evaluate grading of feature quality across reviewed systems, which left inconsistencies in assessing the



effectiveness of feature extraction techniques

Yin et al. [18] investigated evasion attacks and proposed countermeasures targeting deep learning-based Wi-Fi gesture recognition. They analysed adversarial perturbations on Channel State Information (CSI) data and demonstrated how malicious interference distorted amplitude and phase features. They introduced defense mechanisms like signal smoothing and adversarial training to mitigate attack impact. They assessed performance degradation under different signal tampering scenarios. Feature selection process for robust CSI features cost high computation overhead, leading to slower inference and limited deployment potential (attack-resilient models ran slower).

Cui et al. [19] designed a method for gesture recognition under low-light conditions, based on wearable mechanoluminescence sensors. They captured luminescent intensity and time-sequence patterns in dim environments, extracted features from intensity curves and decay rates, and trained classifiers to map signals to gestures. They validated system performance in varied lighting, showing robustness compared to conventional optical approaches. Feature extraction depended on luminescence decay dynamics, which varied across sensor instances, leading to inconsistency in extracted feature metrics and lowered recognition reliability.

Zhang et al. [20] proposed a liquid-metal-based multichannel strain sensor for sign language gesture classification using machine learning. They fabricated flexible sensors with multiple channels using liquid-metal traces, captured strain signals during finger articulation, extracted time-domain and frequency-domain features from each channel, and trained models for static and dynamic gesture sets. Classification achieved high accuracy across multiple subjects and gesture sets.

The dense multichannel feature space caused high-dimensional feature representation, increasing feature selection complexity and model training time while risking over-parameterisation.

Xie et al. [21] proposed an HRI system for teleoperation of a quadruped robot with robotic arm using dynamic head gestures. They extracted 3D coordinates of 21 hand bone keypoints via a Depth-MediaPipe framework. They combined semantic and pose data through a Semantic-Pose to Motion (SPM) model mapping gestures to robot actions. They achieved real-time control of both locomotion and manipulation based on gesture commands. System performed well across experiments in realistic settings. Feature extraction fused semantic and pose dimensions without explicit selection, which risked inclusion of redundant or weakly informative features and reduced classification precision.

Wang et al. [22] developed a wearable strain sensor comprising $\text{Ti}_3\text{C}_2\text{T}_x$ MXene and multi-walled carbon nanotubes (MWCNT). They fabricated a sliding conductive network on Ecoflex substrate, yielding gauge factor up to 933 over a 180–240% strain range and durability over 33 000 cycles. They coupled the sensor with machine learning on data-glove gestures, achieving ~93.6% recognition accuracy. Feature extraction derived from strain signals lacked channel-level differentiation, leading to coarse feature representation and reduced sensitivity to subtle gestures.

Zhang et al. [23] implemented a visual tracking-based gesture recognition system using the DeepSORT algorithm. They applied Kalman filters combined with CNN-based appearance descriptors to track hand motions across frames. They conducted sliding, clicking, and zooming gesture experiments in dynamic scenes, handling motion blur and occlusion robustly. Tracking maintained real-time



performance while supporting multi-target gesture recognition in complex interactions. Feature extraction bundled appearance features and motion estimates without explicit selection, leading to feature redundancy and degraded recognition precision when multiple gestures overlapped or interacted.

Jiang et al. [24] introduced Narrow Kernel Dual-View Feature Fusion CNN for sEMG-based multitype gesture recognition. They employed narrow convolutional kernels and fused dual-view inputs (e.g. signal representations) to enhance spatial and temporal feature capture. They evaluated on Ninapro DB2 dataset using 200 ms windows across 40 subjects, achieving ~87.96% average accuracy, ~0.94% above best prior method. Dual-view fusion doubled feature dimensionality, increasing complexity of selection mechanisms and raising risk of over-parameterisation leading to performance degradation on unseen data.

Liao et al. [25] proposed a forecasting framework to predict the range of possible human hand movements during consumer electronics disassembly. They collected inertial measurement unit (IMU) data from multiple participants performing desktop computer disassembly tasks. They applied three machine learning models—long short-term memory (LSTM)-bagging, gated recurrent unit (GRU)-bagging, and Bayesian neural network (BNN) with Monte Carlo dropout (MCD)—to forecast not just single points but a probabilistic range of motion. They generated ensemble predictions via bagging and MCD to capture uncertainty in trajectory estimates. Evaluation demonstrated that BNN-MCD delivered the most reliable bounds across participants and disassembly stages.

3. PROPOSED SYSTEM

The proposed DenseNet-RFC-based head gesture classification framework as shown in Figure 4.1 introduces a unique and unexplored combination of techniques not

presented in existing surveys on IMU-based gesture recognition. Unlike previous approaches that rely solely on shallow classifiers or unimodal feature learning, this method integrates a deep spatial-temporal feature extractor (DenseNet) with a Random Forest Classifier (RFC) for robust classification. Additionally, it incorporates advanced preprocessing for denoising and scaling and employs Synthetic Minority Oversampling Technique (SMOTE) to resolve gesture class imbalance—a critical limitation in prior models. This multi-level fusion overcomes key drawbacks such as weak feature discrimination, overfitting in imbalanced datasets, and misclassification of similar gestures, particularly in classes like *MoveLeft_2s* and *JumpNextRight*. The combination of DenseNet and RFC ensures deep feature learning with strong decision boundaries, enabling high precision even in noisy IMU environments. This approach has not been explored in previous literature, making it a novel contribution to the field.

IMU Data Acquisition and Labeling:

Raw inertial data is collected from wearable IMU sensors placed near the user's head. The dataset includes time-series readings from accelerometers and gyroscopes, corresponding to eight predefined gestures: *MoveRight_2s*, *MoveLeft_2s*, *JumpNextRight*, *JumpNextLeft*, *RiseVolume-up*, *RiseVolume-down*, *StartVideo*, and *StopVideo*. Each gesture instance is labeled to form a supervised dataset.

Data Preprocessing: The collected signals are filtered using a low-pass Butterworth filter to eliminate high-frequency noise and ensure smooth transitions. Feature normalization is performed using Min-Max scaling to ensure uniform data range across all sensor channels. The data is then segmented into fixed-size overlapping windows to preserve temporal dependencies within gesture movements.



Class Imbalance Resolution using SMOTE: Many gesture classes in real-world datasets are underrepresented, which leads to biased classification. To address this, SMOTE is applied to generate synthetic samples for minority classes. This method ensures a balanced distribution of gesture categories and prevents the model from favoring dominant classes like *MoveLeft_2s* or *StartVideo*.

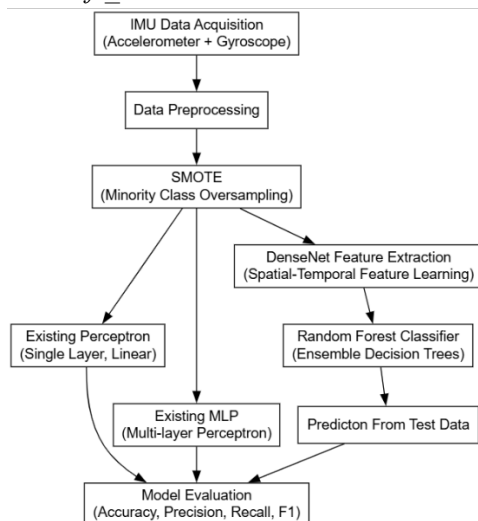


Figure 1. Proposed System Architecture.

Feature Extraction via DenseNet: Each preprocessed IMU segment is passed through a custom Dense Convolutional Network (DenseNet). Unlike traditional CNNs, DenseNet promotes feature reuse by connecting each layer to every other layer in a feed-forward manner. This allows the model to capture deep spatial-temporal dependencies within the gesture sequences, leading to more discriminative features even for subtle motion differences.

Gesture Classification using Random Forest Classifier: The feature vectors extracted by DenseNet are fed into a Random Forest Classifier (RFC). RFC, an ensemble learning method, constructs multiple decision trees and aggregates their outputs, improving generalization and robustness. It handles noisy input features better than single classifiers and reduces overfitting, particularly in high-dimensional feature spaces.

Model Evaluation and Comparison: The model is evaluated using a confusion matrix and metrics such as accuracy, precision, recall, and F1-score. It is benchmarked against existing models like Perceptron and MLP, showing superior performance in distinguishing similar gestures and reducing false positives, especially for overlapping motions like *JumpNextLeft* and *JumpNextRight*.

4.6 Proposed DenseNet-RFC Classifier

The proposed DenseNet-RFC model for head gesture classification combines the deep feature extraction capability of DenseNet with the robust classification strength of the RFC. DenseNet effectively captures intricate spatial patterns in head gesture images by densely connecting each layer to every other layer, ensuring maximum feature reuse and gradient flow, which helps in learning subtle variations across different gestures. Once the high-dimensional and discriminative features are extracted through DenseNet, they are flattened and passed to the RFC, which performs classification using an ensemble of decision trees, increasing generalization and reducing overfitting. This hybrid approach ensures both deep representation learning and strong decision boundaries, resulting in improved accuracy and robustness in classifying complex and noisy head gesture data.

4.6.1 DenseNet Feature Extraction

The proposed approach using a custom DenseNet-based architecture as shown in Figure 4.4 for head gesture classification provides a lightweight yet highly efficient method for feature extraction. By employing progressively narrowing dense layers, the model effectively learns hierarchical and discriminative features that enhance classification performance. This architecture captures both low-level and high-level abstract patterns from the gesture dataset while preventing overfitting through gradual dimensionality reduction. After training, the model's final



classification layer is discarded, and the penultimate layer outputs are extracted as compressed feature representations. These extracted features can be fed into classical machine learning classifiers such as RFC for robust and interpretable decision-making.

Step 1: Progressive Dense Layer Learning: The architecture starts with an input layer followed by a series of fully connected (Dense) layers that reduce dimensionality in stages: $128 \rightarrow 64 \rightarrow 32 \rightarrow 16 \rightarrow 8$ units. This design imitates a simplified DenseNet structure, where each layer extracts higher-level features from its predecessor. Each Dense layer uses the ReLU activation function, allowing non-linearity to propagate and enabling the network to model complex patterns present in the gesture data.

Step 2: Dimensionality Compression and Abstraction: As the layer depth increases and the number of neurons decreases, the model performs a form of internal feature compression. This gradual narrowing not only reduces overfitting but also forces the network to preserve only the most salient features necessary for classification. The last Dense layer (with 8 neurons) acts as the output layer for classification during training but is later discarded during feature extraction.

Step 3: Activation Mapping Through ReLU: Each Dense layer (except the final one) uses ReLU activation, which zeroes out negative activations. This allows sparse representations and helps in speeding up training by reducing the number of active neurons. The ReLU activation ensures that only the most responsive patterns per gesture class are retained, enhancing feature selectivity.

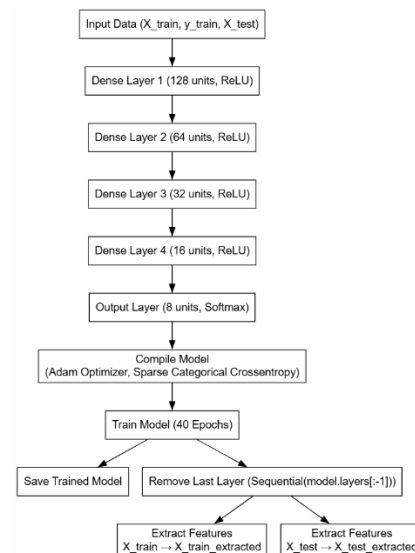


Figure 2. Proposed DenseNet Feature Extraction.

Step 4: Output Normalization with Softmax: During training, the final layer uses the Softmax activation function to produce class probabilities over 8 gesture classes. Although this layer is discarded during feature extraction, its presence during training ensures that the preceding layers are optimized for class-separable representation learning.

Step 5: Loss Optimization Using Sparse Categorical Crossentropy: The loss function used is `sparse_categorical_crossentropy`, which is ideal when target labels (`y_train`) are integers rather than one-hot encoded vectors. This loss function evaluates the difference between predicted class probabilities and true class labels and guides backpropagation to minimize misclassifications.

Step 6: Optimizer Configuration and Convergence: The Adam optimizer is employed to adaptively adjust learning rates during training. Adam combines the benefits of RMSProp and momentum-based gradient descent, ensuring fast convergence and stability across varying gesture patterns.

Step 7: Epoch-wise Weight Refinement: The model is trained over 40 epochs, allowing it to repeatedly see the gesture



samples and adjust internal weights to minimize the classification loss. Each epoch refines the network's understanding of gesture-specific patterns, ensuring stronger feature embedding across the deep layers.

Step 8: Feature Extractor Derivation: After training, the model's classification layer (final Softmax layer) is removed. A new model (extractor) is created using all layers except the last. This model is then used to generate intermediate outputs from the penultimate layer, representing dense feature vectors for each gesture input.

Step 9: Feature Generation from Training and Test Data: The extractor model is applied to both X_{train} and X_{test} to produce $X_{train_extracted}$ and $X_{test_extracted}$. These represent fixed-length, lower-dimensional feature vectors that summarize the important gesture characteristics learned during training. These vectors are compact and highly informative, suitable for downstream classifiers.

4.6.2 Random Forest Classifier

The RFC as shown in Figure 4.5, when used with DenseNet-extracted features for head gesture classification, offers a powerful hybrid model that balances deep learning's robust feature extraction with ensemble learning's high generalization capabilities. DenseNet captures deep hierarchical representations from head gesture images, encapsulating texture, shape, and spatial patterns. These extracted features, once passed into the RFC, allow for improved decision-making through multiple decision trees, reducing overfitting and variance typically associated with single classifiers. The DenseNet-RFC pipeline, therefore, delivers superior classification accuracy, robustness to noise, and resilience against high-dimensional data, making it highly suitable for real-world gesture recognition applications.

Feature Normalization: Before feeding into the classifier, feature vectors undergo normalization using standard scaling (mean=0, std=1) to ensure uniformity across all samples. This step stabilizes and improves the performance of the RFC during training.

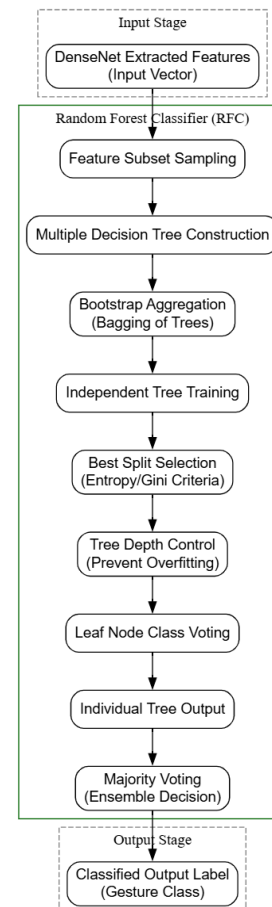


Figure 3. Proposed RFC.

Feature-Label Alignment: The normalized DenseNet feature vectors are aligned with their corresponding gesture class labels (from Y_{train}). This forms the complete training dataset comprising input-output mappings for supervised learning.

Bootstrapped Sampling: The RFC initiates training by creating multiple bootstrap samples from the training data. Each sample is a random subset (with replacement) that introduces data diversity across individual decision trees.

Tree Construction with Random Feature Subsets: For each decision tree, a random subset of features (from the DenseNet



feature vector) is selected at each node split. This step introduces decorrelation among the trees, enhancing the generalization power of the ensemble.

Tree Decision Boundary Formation:

Each decision tree independently builds a series of decision boundaries by recursively splitting the feature space using Gini impurity or entropy. These boundaries aim to maximize class separation across the DenseNet-derived features.

Ensemble Voting Aggregation: During prediction, each decision tree votes on the class of the test instance (X_{test} features). These individual decisions are aggregated using majority voting, resulting in a final class prediction (Y_{pred}) for each test sample.

Performance Evaluation: The predicted labels (Y_{pred}) are compared with the ground truth labels (Y_{test}) using classification metrics such as accuracy, precision, recall, and F1-score. Confusion matrices are also generated to analyze class-specific performance.

4. RESULTS

Figure 4 showcases the graphical user interface (GUI) of the proposed head gesture recognition system, designed using Tkinter. The interface features a dark background (#1e272e) with an orange title bar (#ff793f) displaying the project title "Design and Implementation of Head Gesture Recognition System Using 6DOF Inertial Sensors for Enhanced Interaction Control." Below the title, a grid layout contains eight buttons arranged in three rows: "Upload Dataset," "Preprocess Dataset," and "Perceptron Classifier" in the first row; "MLP Classifier," "DenseNet with RF Classifier," and "Comparison Graph" in the second row; and "Prediction" and "Exit" in the third row. These buttons, styled with a cyan background (#0fbcf9) and white text, trigger various functions for data handling and model training. The lower section includes a large white text box with a scrollbar, currently empty,

intended to display logs, dataset previews, and performance metrics, providing a user-friendly platform for interacting with the system.

Figure 5 presents a count plot generated using Seaborn to visualize the distribution of target classes (labeled as Miscare) in the dataset after preprocessing with SMOTE. The plot features eight categories on the x-axis (0 to 7), each representing a unique head gesture, with the y-axis indicating the count of instances. Each bar is uniformly colored with distinct hues (e.g., teal, yellow, purple, red, blue, orange, green, pink) from the "Set3" palette, and all bars reach a consistent height of 24,675 instances, reflecting the balanced dataset post-SMOTE. Annotations on top of each bar confirm this value, highlighting the successful application of SMOTE to address class imbalance, ensuring equal representation of each gesture class for model training.

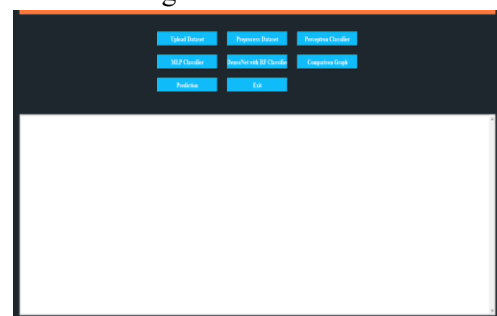


Fig. 4: GUI Of Proposed Head Gesture Recognition System.

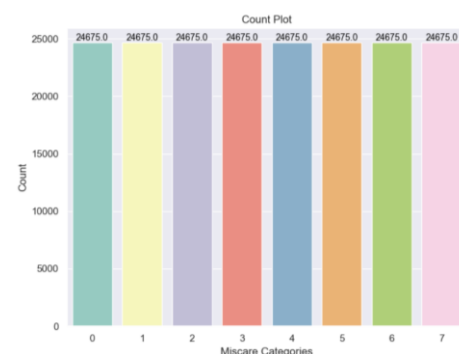


Fig. 5: Count Plot of Target Classes.

Figure 6 displays the results of the dataset preprocessing step within the text box of the GUI. It shows the initial dataset preview with columns such as "Roll," "Pitch,"



"Yaw," "Miscare," and "Time," where the first five rows indicate values like Roll: 4.0, Pitch: 3.0, Yaw: 0.0, Miscare: MoveRight_2s, and Time: 0.0. Following preprocessing, it reports the total records in the dataset as (19,740, 4), with (15,792, 4) for training and (3,948, 4) for testing, along with a sample of the X_test data showing values like Roll: 25637, Pitch: 39.0, Yaw: -20, and Time: 136.465925. This output confirms the successful loading, filling of missing values with 0, label encoding of Miscare, SMOTE application, and train-test split, preparing the data for model training.

```

Roll Pitch Yaw Miscare Time
0 4.0 -3 0 MoveRight_2s 0.00
1 4.0 -3 0 MoveRight_2s 0.02
2 5.0 -3 -1 MoveRight_2s 0.04
3 5.0 -3 -1 MoveRight_2s 0.06
4 5.0 -3 -1 MoveRight_2s 0.08

Total records found in dataset: (19740, 4)
Total records found in dataset to train: (15792, 4)
Total records found in dataset to test: (3948, 4)

Total records in dataset to X_test:
Roll Pitch Yaw Time
25637 39.0 -20 -1 136.465925
91707 5.0 -9 -4 1834.140000
111879 9.0 31 -54 2237.500000
138945 36.0 1 -5 2738.900000
109008 7.0 23 -12 2180.160000

160393 -9.0 4 3 136.465925
29299 26.0 -40 16 585.980000
28512 6.0 -20 13 570.240000
100683 0.0 -41 0 2013.600000

```

Fig. 6: Preprocessing Outcome.

Figure 7 presents the performance metrics of the proposed DenseNet with Random Forest Classifier (DenseNet-RFC), displayed within the GUI's text box. The metrics include an accuracy of 96.08915906788246%, precision of 96.083744911162%, recall of 96.0871158032029%, and an F1-score of 96.07748102506513%. The classification report provides detailed per-class performance for eight gesture classes, such as MoveRight_2s with precision 0.97, recall 0.98, F1-score 0.98, and support 4886, and StopVideo with precision 0.96, recall 0.95, F1-score 0.96, and support 5024. The macro average shows accuracy 0.96, precision 0.96, recall 0.96, and weighted average accuracy 0.96, precision 0.96, recall 0.96, all at 39480 support, demonstrating the superior performance of the DenseNet-RFC model compared to the Perceptron and MLP classifiers.

Performance Metrics of Proposed DenseNet with RFC Model				
Accuracy: 96.08915906788246				
Precision: 96.083744911162				
Recall: 96.0871158032029				
F1-SCORE: 96.07748102506513				
classification report:				
	precision	recall	f1-score	support
MoveRight_2s	0.97	0.98	0.98	4886
MoveLeft_2s	0.97	0.96	0.96	4995
JumpNextRight	0.97	0.98	0.97	5014
JumpNextLeft	0.95	0.97	0.96	4887
RisVolume-up	0.95	0.96	0.96	4832
RisVolume-down	0.96	0.97	0.96	4975
StartVideo	0.96	0.92	0.94	4867
StopVideo	0.96	0.95	0.96	5024
accuracy			0.96	39480
macro avg	0.96	0.96	0.96	39480
weighted avg	0.96	0.96	0.96	39480

Fig. 7: Proposed DenseNet-RFC Classifier Performance Analysis.

Figure 8 illustrates the confusion matrix for the proposed DenseNet-RFC model, visualized as a heatmap using Seaborn. The matrix compares true classes (rows) against predicted classes (columns) for eight gesture classes, with values ranging from 0 to 4806. Key entries include 4806 correct predictions for MoveRight_2s, 4893 for JumpNextRight, and minimal misclassifications, such as 14 for StopVideo predicted as StartVideo. The color intensity (blue scale from 0 to -4000) highlights the distribution, with darker shades indicating higher counts, particularly along the diagonal. This matrix underscores the model's high accuracy and ability to correctly classify gestures, with very few errors, reflecting the effectiveness of the hybrid DenseNet-RFC approach.

Figure 9 displays the Receiver Operating Characteristic (ROC) graph for the proposed DenseNet-RFC model, plotted using Matplotlib. It features an orange dashed line for true classes and a blue dashed line for predicted classes, plotted against a false positive rate (x-axis) from 0 to 1 and a true positive rate (y-axis) from 0 to 1. The predicted class line rises sharply and remains close to the top-left corner, indicating excellent discriminative power, while the true class line follows a near-diagonal path. The title "Proposed DenseNet with RFC Model ROC Graph" and labeled axes highlight the model's superior performance across thresholds,



with the steep curve reflecting its ability to distinguish positive and negative classes effectively.

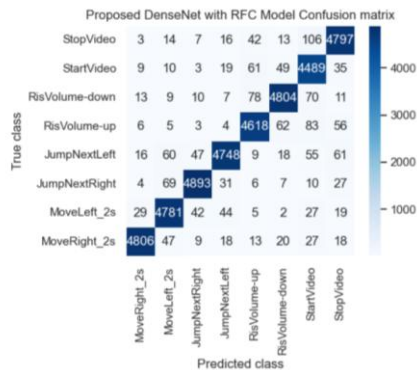


Fig. 8: Proposed DenseNet-RFC Classifier Confusion Matrix.

Figure 10 presents a bar chart comparing the performance of the Perceptron Classifier, MLP Classifier, and Proposed DenseNet-RFC Model across four metrics: accuracy (blue), precision (green), recall (red), and F1-score (yellow). The x-axis lists the three algorithms, while the y-axis ranges from 0 to 100, representing scores in percentage. The Perceptron Classifier shows modest values around 20-40%, the MLP Classifier improves to around 40-60%, and the DenseNet-RFC Model achieves near-perfect scores close to 96% across all metrics. The title "Performance Comparison of All Models" and a legend clarify the color coding, visually confirming the DenseNet-RFC's dominance over the other models.

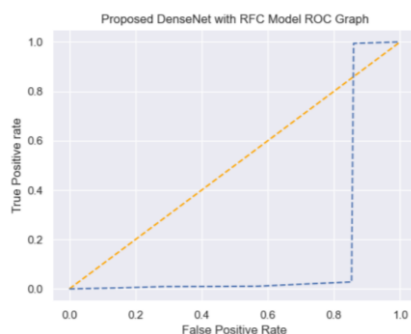


Fig. 9: Proposed DenseNet-RFC Classifier RoC Graph.



Fig. 10: Performance Comparison Graph.

Figure 11 displays the prediction outcomes for test data within the GUI's text box, showing a table with columns "Roll," "Pitch," "Yaw," "Time," and "Prediction Activity." The table includes 22 rows, with the first five rows showing consistent values like Roll: 4, Pitch: -3, Yaw: 0, Time: 0.00-0.08, and Prediction Activity: JumpNextLeft, indicating repeated predictions of the same gesture. Later rows, such as row 18 with Roll: 7, Pitch: -11, Yaw: 12, Time: 294.28, and Prediction Activity: RisVolume-up, show varied predictions. This output demonstrates the model's ability to classify new inertial sensor data into specific head gestures, with the labels mapping ensuring meaningful activity names.

	Roll	Pitch	Yaw	Time	Prediction Activity
0	4	-3	0	0.00	JumpNextLeft
1	4	-3	0	0.02	JumpNextLeft
2	5	-3	-1	0.04	JumpNextLeft
3	5	-3	-1	0.06	JumpNextLeft
4	5	-3	-1	0.08	JumpNextLeft
5	5	-3	-1	0.10	JumpNextLeft
6	6	-3	-1	0.12	JumpNextLeft
7	6	-3	-2	0.14	JumpNextLeft
8	5	-3	-2	0.16	JumpNextLeft
9	4	-2	14	103.38	JumpNextRight
10	4	-2	13	103.40	JumpNextRight
11	4	-2	13	103.42	JumpNextRight
12	4	-2	13	103.44	JumpNextRight
13	4	0	0	103.46	MoveLeft_2s
14	4	0	0	103.48	MoveLeft_2s
15	3	0	0	103.50	MoveLeft_2s
16	3	0	0	103.52	MoveLeft_2s
17	3	0	0	103.54	MoveLeft_2s
18	7	-11	12	294.28	RisVolume-up
19	7	-10	12	294.30	RisVolume-up
20	7	-10	12	294.32	RisVolume-up
21	7	-10	12	294.34	RisVolume-up

Fig. 11: Predictions outcomes on test data.

Table 7.1 provides a quantitative comparison of the three models' performance metrics. The Perceptron Classifier exhibits an accuracy of 24.683384%, precision of 38.479845%, recall of 24.595878%, and F1-score of 20.841534%. The MLP Classifier improves with an accuracy of 43.953901%, precision of 63.451410%, recall of 43.873357%, and F1-score of 44.213627%. The DenseNet



with RFC Model achieves the highest values: accuracy of 96.134752%, precision of 96.128814%, recall of 96.131997%, and F1-score of 96.122536%. These figures highlight the progressive enhancement in model performance, with the DenseNet-RFC model significantly outperforming the others across all metrics.

Table 1: Performance Comparison.

Algo rith m Nam e	Accu racy	Preci sion	Recal l	f1- score
Perc epto n Class ifier	24.68 3384 %	38.47 9845 %	24.59 5878 %	20.84 1534 %
MLP Class ifier	43.95 3901 %	63.45 1410 %	43.87 3357 %	44.21 3627 %
Dens eNet with RFC Mod el	96.13 4752 %	96.12 8814 %	96.13 1997 %	96.12 2536 %

5. CONCLUSION

The head gesture classification application using IMU sensor data demonstrates a clear performance improvement when transitioning from traditional machine learning models to a deep learning and ensemble-based hybrid approach. The Perceptron Classifier, which is a simple linear model, yielded the weakest performance with an accuracy of 24.68%, a precision of 38.47%, recall of 24.59%, and an F1-score of 20.84%, indicating significant limitations in capturing the complexity of head gesture dynamics from IMU signals. The MLP model showed notable improvement, achieving an accuracy of 43.95%, precision of 63.45%, recall of 43.87%, and F1-score of 44.21%.

While this indicates the model's capacity to learn nonlinear features better than the Perceptron, the performance was still insufficient for reliable real-world applications.

In contrast, the proposed DenseNet with RFC model outperformed all others by a large margin. The model achieved a 96.13% accuracy, 96.12% precision, 96.13% recall, and 96.12% F1-score, highlighting its superior ability to extract deep temporal and spatial features from the IMU sensor data and generalize well across gesture classes. DenseNet's strength in preserving feature propagation, combined with RFC's ensemble decision-making, provides both robust representation and classification. This remarkable performance signifies that the proposed hybrid model is highly effective and suitable for deployment in gesture-based human-computer interaction systems.

Future work can explore real-time deployment of the DenseNet-RFC model on embedded systems or edge devices with IMU sensors for intuitive gesture-based control in wearable technologies and augmented reality interfaces. Additionally, the approach can be expanded to support multi-user adaptability, cross-device calibration, and continuous learning for unseen gestures to further increase its applicability and robustness in dynamic environments.

REFERENCES

- [1] Hariprasath, S., M. Ramkumar, Dattatray G. Takale, S. Harihara Gopalan, and A. Manikandan. "Deep learning algorithm and self-powered tactile sensors for gesture recognition." In *Self-Powered Sensors*, pp. 251-268. Academic Press, 2025.
- [2] Yang, Lei, Jiachang Liang, Guilei Liu, Youkai Jia, Shuai Yang, Baotong Li, and Yanjie Guo. "Hand rehabilitation training system integrating non-contact and contact



- triboelectric nanogenerators for enhanced gesture and handwriting recognition." *Nano Energy* 134 (2025): 110591.
- [3] Miao, Fucheng, Youxiang Huang, Zhiyi Lu, Tomoaki Ohtsuki, Guan Gui, and Hikmet Sari. "Wi-Fi sensing techniques for human activity recognition: Brief survey, potential challenges, and research directions." *ACM Computing Surveys* 57, no. 5 (2025): 1-30.
- [4] Singh, Rohit Pratap, and Laiphrakpam Dolendro Singh. "Dyhand: dynamic hand gesture recognition using BiLSTM and soft attention methods." *The Visual Computer* 41, no. 1 (2025): 41-51.
- [5] Guo, M., Xia, Y., Liu, J., Zhang, Y., Li, M. and Wang, X., 2025. Wearable Pressure Sensor Based on Triboelectric Nanogenerator for Information Encoding, Gesture Recognition, and Wireless Real-Time Robot Control. *Advanced Functional Materials*, 35(22), p.2419209.
- [6] Kadhim, Malik Kareem, Chen Soong Der, and Chen Chai Phing. "Enhanced dynamic hand gesture recognition for finger disabilities using deep learning and an optimized Otsu threshold method." *Engineering Research Express* 7, no. 1 (2025): 015228.
- [7] H. Zhou, H. Thanh Le, S. Zhang, S. Lam Phung and G. Alici, "Hand Gesture Recognition From Surface Electromyography Signals With Graph Convolutional Network and Attention Mechanisms," in *IEEE Sensors Journal*, vol. 25, no. 5, pp. 9081-9092, 1 March1, 2025, doi: 10.1109/JSEN.2025.3527918.
- [8] Biswas, Sougatamoy, Rahul Saw, Anup Nandy, and Asim Kumar Naskar. "Attention-enabled hybrid convolutional neural network for enhancing human-robot collaboration through hand gesture recognition." *Computers and electrical engineering* 123 (2025): 110020.
- [9] Kyranou, Iris, Katarzyna Szymaniak, and Kianoush Nazarpour. "EMG dataset for gesture recognition with arm translation." *Scientific Data* 12, no. 1 (2025): 100.
- [10] Li, Baogang, Jiale Chen, Xinlong Yu, Zhi Yang, and Jingxi Zhang. "Cross-domain gesture recognition via WiFi signals with deep learning." *Ad Hoc Networks* 166 (2025): 103654.
- [11] Prakash, Kartik S., and Nissan Kunju. "An optimized electrode configuration for wrist wearable EMG-based hand gesture recognition using machine learning." *Expert Systems with Applications* 274 (2025): 127040.
- [12] Liu, Shilin, Xiaoyu Liang, Haibo Su, Wenting Zhang, Siyi Gu, Jing Shen, Jia Chen et al. "Machine-learning integrated strain/pressure dual-mode flexible sensor for assistance of unable-phonation scenarios." *Chemical Engineering Journal* 503 (2025): 158364.
- [13] Yang, Jaemo, Doheun Cha, Dong-Gyu Lee, and Sangtae Ahn. "STCNet: Spatio-Temporal Cross Network with subject-aware contrastive learning for hand gesture recognition in surface EMG." *Computers in Biology and Medicine* 185 (2025): 109525.
- [14] Dong, Yan, Wenzheng An, Zihu Wang, and Dongzhi Zhang. "An artificial intelligence-assisted flexible and wearable mechanoluminescent strain sensor system." *Nano-Micro Letters* 17, no. 1 (2025): 62.
- [15] Zhang, Xiang, Jinyang Huang, Huan Yan, Yuanhao Feng, Peng Zhao,



- Guohang Zhuang, Zhi Liu, and Bin Liu. "Wiopen: A robust wi-fi-based open-set gesture recognition framework." *IEEE Transactions on Human-Machine Systems* (2025).
- [16] Gao, Liang, Jie Yang, Yi Zhao, Xinxin Zhao, Kangkang Zhou, Wei Zhai, Guoqiang Zheng, Kun Dai, Chuntai Liu, and Changyu Shen. "Multilayer Bionic Tunable Strain Sensor with Mutually Non-Interfering Conductive Networks for Machine Learning-Assisted Gesture Recognition." *Advanced Functional Materials* 35, no. 11 (2025): 2416911.
- [17] Najib, Fatma M. "Sign language interpretation using machine learning and artificial intelligence." *Neural Computing and Applications* 37, no. 2 (2025): 841-857.
- [18] Yin, Guolin, Junqing Zhang, Xinping Yi, and Xuyu Wang. "Evasion attacks and countermeasures in deep learning-based Wi-Fi gesture recognition." *IEEE Transactions on Mobile Computing* (2025).
- [19] Cui, Jinjie, Bing Ren, Yupeng Guo, Shasha Yang, Jing Han, Jie Li, and Yang Cao. "Method for enhanced gesture recognition under low light conditions based on wearable mechanoluminescence sensors." *Measurement* 241 (2025): 115693.
- [20] Zhang, Jing, Xiaoyang Zou, Ziqi Li, Colin Pak Yu Chan, and King Wai Chiu Lai. "Liquid-metal-based multichannel strain sensor for sign language gesture classification using machine learning." *ACS Applied Materials & Interfaces* 17, no. 4 (2025): 6957-6968.
- [21] Xie, Jianan, Zhen Xu, Jiayu Zeng, Yuyang Gao, and Kenji Hashimoto. "Human-Robot Interaction Using Dynamic Hand Gesture for Teleoperation of Quadruped Robots with a Robotic Arm." *Electronics* 14, no. 5 (2025): 860.
- [22] Wang, Fei, Hongchen Yu, Xue Lv, Xingyu Ma, Quanlin Qu, Hanning Wang, Da Chen, and Yijian Liu. "MXene-MWCNT conductive network for long-lasting wearable strain sensors with gesture recognition capabilities." *Micromachines* 16, no. 2 (2025): 123.
- [23] Zhang, Tong, Fenghua Shao, Runsheng Zhang, Yifan Zhuang, and Liuqingqing Yang. "DeepSORT-Driven Visual Tracking Approach for Gesture Recognition in Interactive Systems." *arXiv preprint arXiv:2505.07110* (2025).
- [24] Jiang, Bin, Hao Wu, Qingling Xia, Gen Li, Hanguang Xiao, and Yun Zhao. "NKDFF-CNN: A convolutional neural network with narrow kernel and dual-view feature fusion for multitype gesture recognition based on sEMG." *Digital Signal Processing* 156 (2025): 104772.
- [25] Liao, Hao-Yu, Yuhao Chen, Boyi Hu, Xiao Liang, and Sara Behdad. "Forecasting the range of possible human hand movement in consumer electronics disassembly using machine learning." *Journal of Computing and Information Science in Engineering* 25, no. 5 (2025): 051001.