Window-Free IMU-Based Classification of Stair-Climbing Wheelchair Activities Using Machine Learning and Adaptive Boosting

Pharan Chawaphan
Department of Mechatronics Engineering, Rajamangala University of Technology Thanyaburi, Pathum Thani, Thailand
pharan_c@mail.rmutt.ac.th

Dechrit Maneetham
Department of Mechatronics Engineering, Rajamangala University of Technology Thanyaburi, Pathum Thani, Thailand
dechrit_m@rmutt.ac.th (corresponding author)

Padma Nyoman Crisnapati
Department of Mechatronics Engineering, Rajamangala University of Technology Thanyaburi, Pathum Thani, Thailand
crisnapati@rmutt.ac.th

Received: 15 October 2025 | Revised: 1 December 2025, 28 December 2025, and 5 January 2026 | Accepted: 6 January 2026

 DOI: https://doi.org/10.48084/etasr.15555

ABSTRACT

To ensure safety, autonomy, and context-aware control, reliable state recognition is still a challenge for stair-climbing wheelchairs, which offer mobility in environments with steps, curbs, and landings. This study uses instantaneous Inertial Measurement Unit (IMU) data to develop a simplified, window-free method for classifying wheelchair stair-related activities. Instead of sliding-window preprocessing or temporal sequence modeling, this study uses an 18-channel feature set that includes orientation, gyroscope, accelerometer, magnetometer, linear acceleration, and gravity signals. Stratified evaluation and several metrics, such as accuracy, macro-F1, Matthews Correlation Coefficient (MCC), ROC-AUC, and per-class precision-recall, were used to systematically benchmark eight classifiers: Multinomial Logistic Regression (MLR), Gaussian Naïve Bayes (GNB), K-Nearest Neighbors (KNN), Decision Tree (DT), Random Forest (RF), RBF-kernel Support Vector Classifier (SVC), a compact Multilayer Perceptron (MLP), and an adaptive CatBoost configuration. The results show that CatBoost performed almost flawlessly (Accuracy = 0.999, MCC = 0.999), closely followed by compact MLP. RF, KNN, and SVC formed a solid middle tier. In a window-free regime, feature importance analysis showed that instantaneous gyroscope and linear acceleration made very little contribution, while orientation and magnetometer channels were found to be the most crucial features. These results show that accurate and computationally efficient recognition of stair-climbing wheelchair states is possible by features driven by posture and heading. The proposed method facilitates low-latency embedded deployment and identifies areas for future development, such as lightweight temporal enhancements, angle encoding, and magnetometer calibration.

Keywords-stair-climbing wheelchair; Inertial Measurement Unit (IMU); Human Activity Recognition (HAR); window-free classification; machine learning

I. INTRODUCTION

Stair-climbing wheelchairs allow people with disabilities to move around in places that still have steps, curbs, and landings [1]. For safety interlocks, autonomous assistance, and context-aware control, it is very important to reliably tell what state and phase of a maneuver the device is in, such as approach, ascent, descent, or landing [2]. Inertial Measurement Units (IMUs) are a good choice for this because they are small, inexpensive, and already on many platforms [3]. Many studies on Human Activity Recognition (HAR) have shown that IMU signals can accurately identify different types of movement [4]. However, most pipelines use sliding-window preprocessing and temporal feature engineering, which are slow and harder to compute [5]. Recent work has also focused on anti-tip and anti-roll systems to improve the safety and stability of these devices [6].

Classical HAR benchmarks (such as accelerometer/gyroscope at 50 Hz) did very well when using windowed time- and frequency-domain data with Support Vector Machines (SVM), Random Forest (RF), and other classifiers [7]. Later surveys show a move toward deep sequence models, such as convolutional, recurrent, and transformer-based architectures, that learn temporal patterns directly from multichannel IMU streams [8]. At the same time, wearable studies on exoskeletons that focus on gait and stair negotiation focus on event detection and mode transitions using tiny sensor sets and real-time restrictions, commonly using thresholding or hidden Markov models [9]. These approaches jointly substantiate IMU-based recognition but focus on human lower-limb motion or phone-mounted sensing rather than the kinematics of a stair-climbing wheelchair [10].

There is not much research on stair-climbing wheelchairs and their differences [11]. Many studies focused on how to design and operate climbing devices, whereas perception components often use eyesight to tell the difference between stairs and floors [12]. Most reports that use IMUs on wheelchair platforms follow the mainstream HAR paradigm, which includes windowed preprocessing, class balancing, and either boosted trees or compact neural networks [13]. However, most of them do not examine which sensing modalities (orientation, gravity, magnetometer, accelerometer, gyro, linear acceleration) are most useful for wheelchair maneuvers [14]. Consequently, two deficiencies remain: (i) the absence of real-time, windowless classifiers appropriate for ultra-low-latency embedded deployment, and (ii) restricted, device-specific interpretability that connects sensor significance to the physics of stair negotiation [15].

This study addresses these deficiencies by introducing a streamlined, tabular pipeline for detecting stair-climbing wheelchair activity, compatible with single-sample IMU vectors (18 channels: orientation, gyroscope, accelerometer, magnetometer, linear acceleration, gravity) and devoid of sliding windows or sequence encoders [16]. A comparison of eight typical models, Multinomial Logistic Regression (MLR), Gaussian Naïve Bayes (GNB), K-Nearest Neighbors (KNN), Decision Tree (DT), RF, RBF-kernel SVC, a compact Multilayer Perceptron (MLP), and an adaptive CatBoost configuration was performed [17]. This approach uses a consistent stratified split and a complete set of metrics, such as accuracy, macro-F1, Matthews Correlation Coefficient (MCC), ROC-AUC, confusion matrices, and per-class precision-recall [18]. The boosted-tree and compact neural baselines performed almost at the highest level, even without temporal aggregation [19]. This demonstrates that positional and directional cues alone can consistently differentiate various wheelchair stair-related states [20].

The study improves both the accuracy and the ability to understand devices. According to CatBoost's feature-importance analysis, orientation and magnetometer channels are the most essential. In the window-free regime, the gravity components and instantaneous gyroscope/linear-acceleration signals were less relevant [21]. This hierarchy is comparable to how tracked or wheeled stairs function. The chassis heading and pitch/roll relative to gravity determine how to get on, go up, go down, and land. The results show effective strategies for embedded systems, such as minimizing the number of features to focus on the most important channels. They also show how to improve in the future, including using temporal characteristics from dynamic sensors when latency budgets allow.

In conclusion, this research demonstrates that instantaneous window-free IMU classification can equal or exceed windowed baselines for stair-climbing wheelchair applications, all while being computationally economical and easily comprehensible. Researchers and practitioners developing perception stacks for stair-climbing mobility devices can utilize a comprehensive, rigorously controlled model comparison, calibration-sensitive evaluation, and modality-specific attribution as a singular information source.

II. RESEARCH METHODOLOGY

A. Data Acquisition

A stair-climbing IMU dataset [22] was obtained, containing 5,873 instantaneous IMU sensor data during real stair-climbing wheelchair operation. The sensor was sampled at 50 Hz. The total length of this recording is approximately 117 seconds, with the following class distribution across six wheelchair activities: 1,512 samples for Class 0, 1,851 samples for Class 1, 622 samples for Class 2, 632 samples for Class 3, 640 samples for Class 4, and 616 samples for Class 5. There are 18 continuous sensor channels in the feature set: orientation (o_x, o_y, o_z), gyroscope (g_x, g_y, g_z), acceleration (a_x, a_y, a_z), magnetometer (m_x, m_y, m_z), linear acceleration (l_x, l_y, l_z), and gravity (gr_x, gr_y, gr_z). A short examination (summary statistics and counts of missing values) was performed to ensure the numeric ranges are correct, find possible problems, and count the NaNs that need to be fixed during preprocessing.

IMU readings have inherent sensor noise and bias drift. This study used the BNO055 sensor, which is specified to have ±0.5° RMS orientation accuracy and up to 0.7–0.9° deviation under magnetically disturbed conditions according to the manufacturer.

B. Preprocessing

To normalize the class labels, one was subtracted from *label_id*, giving a range of labels from 0 to 5. This relabeling ensures that training and evaluation methods are the same, especially for the One-vs-Rest (OvR) binarization utilized in ROC and precision-recall analyses [23]. Feature scaling was not performed globally because different model families need different treatments. Instead, scaling was performed inside per-model pipelines, where it makes sense.

C. Data Splitting

To keep the original class proportions, the dataset was split into training and test sets using a stratified 80:20 split [24]. Stratification keeps the class balance in both subgroups and offers a better estimate of generalization. The sorted unique label set is stored to provide consistent class ordering when converting labels to binary for OvR metrics.

D. Model Pipelines

A distinct pipeline was used for each baseline algorithm: MLR, GNB, KNN (k=5), DT (depth=6), RF (100 trees, depth=6), RBF-kernel SVC (probability enabled), and a tiny MLP (32–16). Median imputation was used to deal with NaNs at the start of each pipeline. Standardization was only used for models that work better with scaled inputs, such as MLR, LNN, SVC, MLP, and GNB. Tree-based models utilize the imputed features without scaling due to their scale invariance. To avoid leaking, a gradient-boosted DT model (CatBoost) was trained with early halting on a validation subset taken from the training data (not on the test set). The design used multi-class loss, a moderate learning rate, regulated depth, and automatic class weighting to fix the imbalance. The final model was tested on the unseen test set after early stopping, choosing the optimal iteration based on validation performance. For a direct comparison with the baseline models, the same set of metrics and plots of accuracy, macro-F1, MCC, confusion matrix, OvR ROC-AUC, and per-class ROC/PR curves was used [25, 26].

E. Training

Each baseline pipeline is only fitted on the training split; thus, the model parameter selection is not affected by any information from the test data. Hyperparameters are maintained low so that they can be used as benchmarks for stronger models without being too tweaked. CatBoost uses 2000 boosting iterations with depth-6 symmetric trees, a learning rate of 0.05, multi-class log-loss, automatic class weighting to address mild imbalance, and validation-based overfitting detection (od_type='Iter', od_wait=50) with use_best_model=True. The other baselines follow the following configurations: MLR with $\ell 2$ regularization, BGN with class-conditional Gaussians, KNN with k = 5, a depth-6 DT, an RF with 100 trees of depth-6, an RBF-kernel SVC with probability estimates enabled, and a compact MLP with a 32–16 hidden-layer structure.

Several methods were employed to prevent overfitting. The dataset was split into train, validation, and test sets, and only the training and validation sets were used in model fitting. For CatBoost, a separate validation pool (val_pool), and enable overfitting detection, od_type = 'Iter', od_wait = 50, use_best_model = True was used to stop early when the validation loss stops improving, and keep the best iteration on the validation set. Depth was limited to 6, and the learning rate was moderate at 0.05, which also constrains model capacity. Finally, all performance results come exclusively from the held-out 20% test set, which the model never sees during training or early-stopping.

F. Prediction and Evaluation

Predictions made on the held-out test set give class assignments and, for models that know about probabilities, class-membership probabilities. Confusion matrices are used to determine how errors are spread out among classes. For models that show calibrated or at least monotonic probabilities through predict_probability, OvR ROC curves, and precision-recall curves are used for each class using label binarization. The OvR ROC-AUC is presented as a short probabilistic summary.

Fig. 1. Research overview.

G. Model Formulations

Let $x \in R^{18}$ denote the IMU feature vector and $y \in \{0, \dots, 5\}$ the class label. Multiclass models produce a vector of real-valued scores $z(x)$, which are transformed into class probabilities using the softmax function (1), normalizing the scores to form a valid probability simplex. MLR assigns a linear score to each class (2). Softmax (3) maps the score vector to class probabilities, and prediction (4) picks the class with the highest probability [27]. The cross-entropy loss can be minimized to learn parameters, with optional $\ell 2$ regularization applied to prevent the model from overfitting (5). This results in linear decision boundaries within the input space. GNB presupposes that the features are conditionally independent given the class, and that each follows a Gaussian distribution inside a class (6), with a class prior denoted as πk (7). Based on these assumptions, the (unnormalized) log-posterior is the sum of the log prior and the Gaussian log-likelihoods for each feature (8). Prediction selects the class that optimizes this log-posterior [28].

$$\text{softmax}(z)_k = \frac{e^{z_k}}{\sum_{j=0}^{5} e^{z_j}} \quad (1)$$

$$z_k(x) = w_k^T x + b_k \quad (2)$$

$$p(y = k \mid x) = \frac{e^{z_k(x)}}{\sum_{j=0}^{5} e^{z_j(x)}} \quad (3)$$

$$\hat{y} = \underset{k}{arg\,max} p(y = k \mid x) \quad (4)$$

$$\mathcal{L}_{\text{LR}} = -\sum_i \log\, p(y_i \mid x_i) + \lambda \sum_{k=0}^{5} \|w_k\|_2^2 \quad (5)$$

$$p(x_j \mid y = k) = \mathcal{N}(x_j; \mu_{kj}, {\sigma_{kj}}^2) \quad (6)$$

$$p(y = k) = \pi_k \quad (7)$$

$$\log\, p(y = k \mid x) \propto \log\, \pi_k - \frac{1}{2}\sum_j [\log\,(2\pi {\sigma_{kj}}^2) + \frac{(x_j - \mu_{kj})^2}{\sigma_{kj}^2}] \quad (8)$$

KNN sorts samples based on how close they are in feature space, using Euclidean distance (9). The predicted class is the most common label among the k closest training points (10), and the empirical class probability is the number of neighbors in each class (11). A distance-weighted version uses inverse-distance weights instead of unit votes to reduce the effect of neighbors that are far away [29]. A DT divides the feature space into smaller parts by picking splits that make the nodes less impure. The Gini index (12) measures how impure something is. The weighted impurity of the left and right children is minimized by choosing each split (13). The criterion for selecting the best split is defined by the minimization process (14). Leaves anticipate either the most common class or the class distribution that is actually present at that leaf [30].

$$d(x, x_i) = \| x - x_i \|_2 \quad (9)$$

$$\hat{y} = \underset{k}{arg\,max} \sum_{x_i y_i \in N_k(x)} \mathbf{1}\{y_i = k\} \quad (10)$$

$$\hat{p}(y = k \mid x) = \frac{1}{k}\sum_{x_i y_i \in N_k(x)} \mathbf{1}\{y_i = k\} \quad (11)$$

$$G = 1 - \sum_k p_k^2 \quad (12)$$

$$(j^*, t^*) = \underset{j,t}{arg\,min} \frac{n_L}{n} G_L(j,t) + \frac{n_R}{n} G_R(j,t) \quad (13)$$

$$\hat{p}(y = k \mid x) = \frac{1}{B}\sum_{b=1}^{B} \hat{p}_b(y = k \mid x) \quad (14)$$

An RF comprises a group of DTs by using bootstrap samples and feature subsampling. The average of the class probabilities from all the trees is used to find the predicted class (15). Averaging reduces the variance compared to a single tree and usually makes generalization better [31]. SVC is a large-margin classifier. With an RBF kernel, a kernelized decision function that is a weighted sum over support vectors (16) gives non-linear decision boundaries. The parameters are determined by minimizing the soft-margin objective, balancing the margin size with the number of hinge-loss violations (17). Platt scaling the margins (18) is the most common way to obtain the probability [32].

TABLE I. NOMENCLATURE

Symbol	Description
x	IMU feature vector (18-dimensional)
x_i	i^{th} component of feature vector x
y	Ground-truth class label $y \in \{0,1,2,3,4,5\}$
$\hat{y}$	Predicted class label
$z(x)$	Raw score or logit vector output of the model
$p(y = k \mid x)$	Predicted probability that sample x belongs to class k
K	Number of classes (6)
w_k	Weight vector for class k (Logistic Regression)
b_k	Bias term for class k
l_2	Regularization coefficient
π_k	Prior probability of class k (Gaussian Naïve Bayes)
μ_{kj}	Mean of feature j for class k
σ_{kj}^2	Variance of feature j for class k
k	Number of neighbors in kNN (k=5)
$d(x_i, x_j)$	Euclidean distance between feature vectors
$g(t)$	Impurity measure (e.g., Gini index)
T	Set of trees in an ensemble model (RF or CatBoost)
$h_t(x)$	Weak learner/tree added at boosting iteration t
$f_t(x)$	Model output after t boosting stages
α_t	Learning rate or tree weight in boosting
ξ_i	Slack variable in SVC optimization
C	Penalty parameter controlling SVC margin error
$K(x_i, x_j)$	Kernel function (RBF kernel)
γ	RBF kernel coefficient
θ	Neural network parameters (weights and biases)
$a^{(l)}$	Activation output of layer l
$W^{(l)}$	Weight matrix of layer l
$b^{(l)}$	Bias vector of layer l
ReLU(·)	Rectified Linear Unit activation function
Softmax(·)	Softmax probability normalization
L	Cross-entropy loss function
∇L	Gradient of the loss with respect to model parameters

$$\hat{y} = \underset{k}{arg\,max} \hat{p}(y = k \mid x) \quad (15)$$

$$f(x) = \sum_{i=1}^{n} \alpha_i y_i K(x, x_i) + b, \quad K(x, x_i) = \exp(-\gamma \| x - x_i \|_2^2) \quad (16)$$

$$\underset{w,b,\xi}{min} \frac{1}{2}\|w\|_2^2 + C\sum_i \xi_i \quad \text{s.t.} \quad y_i(w^T \varphi(x_i) + b) \geq 1 - \xi_i$$

$$\xi_i \geq 0 \quad (17)$$

$$p(y = 1 \mid f) = \frac{1}{1 + \exp(Af + B)} \quad (18)$$

The MLP uses affine layers and ReLU activations to make a non-linear map. The first hidden layer and activation are shown in (19), and (20) shows the second layer and logits. Softmax changes logits into class probabilities. Training reduces the multiclass cross-entropy (21), leading to trained non-linear decision boundaries that are flexible [33]. CatBoost creates an additive model of shallow, symmetric (oblivious) trees by fitting trees to the negative gradients (residuals) of the multiclass log-loss one at a time. The additive update works as in (22). Softmax can be applied to the final score vector to obtain probabilities (23). The optimized objective is the multiclass log-loss (24), and its gradient with respect to the scores simplifies to "prediction minus one-hot target" (25). Early stopping finds the best number of trees by using a validation set. Optional class weights change the effective loss for each class [34]. Table I describes all symbols in equations.

$$h = \mathrm{ReLU}(W_1 x + b_1) \quad (19)$$

$$z = W_2 h + b_2, p(y = k \mid x) = \mathrm{softmax}(z)_k \quad (20)$$

$$\mathcal{L}_{\mathrm{MLP}} = -\sum_i \log\, p(y_i \mid x_i) \quad (21)$$

$$F^0(x) = 0, \quad F^{(m)}(x) = F^{(m-1)}(x) + \nu f^{(m)}(x) \quad (22)$$

$$p(y = k \mid x) = softmax(F(x))_k \quad (23)$$

$$\mathcal{L}_{\mathrm{CAT}} = -\sum_i \log\, p(y_i \mid x_i) \quad (24)$$

$$g_{ik} = \frac{\partial \mathcal{L}_{\mathrm{CAT}}}{\partial F_k(x_i)} = p_{ik} - y_{ik} \quad (25)$$

III. RESULTS AND DISCUSSION

Figure 2 shows eight confusion matrices for a six-class IMU activity classification challenge. Rows show real classes, and columns show anticipated classes. Darker diagonal cells indicate right assignments, whereas off-diagonal ones indicate wrong assignments. Visual inspection shows clear changes in diagonal dominance amongst models, offering a comparison of how classifiers behave on the same data and split. The matrices show a three-tier performance pattern. Adaptive CatBoost and the MLP (32–16) make almost perfect diagonals at the top tier, indicating very few errors in all classes. The middle layer, which includes RF (100, depth = 6), kNN (k = 5), and SVC (RBF), has strong diagonals with minor clusters of off-diagonal errors, indicating excellent but not perfect accuracy. The second layer, which includes MLR, GNB, and a single DT (depth = 6), shows a lot more spread away from the diagonal, indicating that class confusions happen a lot.

These trends are consistent with the modeling assumptions and capabilities. MLR requires linear decision boundaries, making it sensitive to class structures that are not linearly separable in the 18-dimensional sensor space. GNB depends on conditional independence and Gaussian feature distributions. When IMU channels are correlated, these assumptions are broken, increasing error rates. A single shallow DT usually has considerable variance and local overfitting, making it perform unevenly across classes. On the other hand, RF reduces variance by bagging and feature subsampling, KNN uses the structure of the neighborhood, and SVC with an RBF kernel finds smooth nonlinear bounds. The MLP learns how to do hierarchical nonlinear transformations, and CatBoost's gradient-boosted symmetric trees are great at modeling high-order interactions and dealing with class imbalance, which is why the diagonals are so close to flawless.

The errors imply that some class pairs, especially those with intermediate labels, are harder to tell apart when models are weaker. This is probably due to similar feature distributions or motion patterns, and can be reduced by adding information from the time and frequency domains (such as windowed statistics and spectrum energy), fine-tuning class weights or decision thresholds, and using probability calibration where judgments downstream depend on confidence. When accuracy is the most important thing, gradient-boosted trees or compact neural networks are the best choices. When model simplicity or interpretability is important, RF is the best choice, as it offers performance and low complexity.

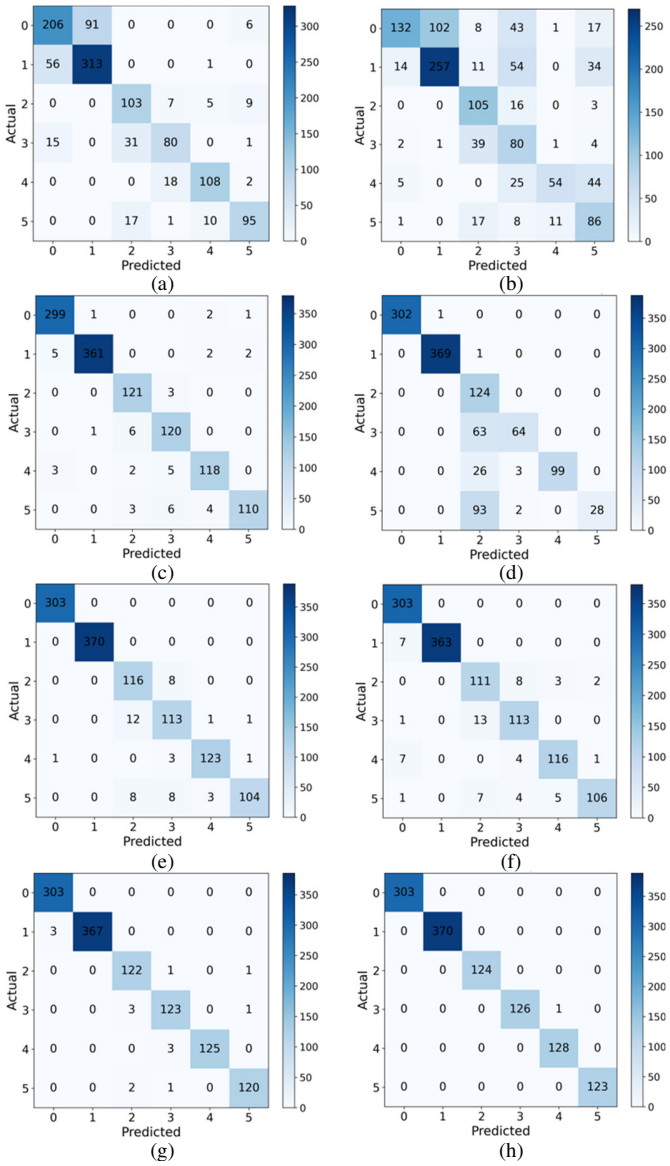

Fig. 2. Confusion matrices: (a) MLR, (b) GNB, (c) KNN, (d) DT, (e) RF, (f) SVC (RBF), (g) MLP, (h) Adaptive CatBoost.

In Figure 3, the legend gives the AUC for each class, and each panel shows the true-positive rate compared to the false-positive rate for the six classes. Curves close to the upper-left corner and AUC values close to 1.0 mean that the model can tell the difference between different thresholds very well. In general, the models can be put into three groups based on how well they work. As shown in Figure 3(g, h), the ROC curves for all classes (AUC = 0.998–1.000) for Adaptive CatBoost and MLP (32–16) are almost flawless, indicating that the separation is very strong and not affected by the choice of threshold. As shown in Figure 3 (e, c, f), RF (100x depth-6), KNN (k=5), and SVC (RBF) have curves that are densely grouped in the top-left corner, with class-wise AUC usually >0.99. This means that the classes are strong but not quite as evenly separated. The lower-tier MLR, DT (depth-6), and GNB, shown in Figure 3(a, d, b), demonstrate a wider spread and a lower AUC for some

classes. For example, the GNB and the DT have curves that bend away from the top-left corner, which means they are less able to tell the difference between classes based on their modeling assumptions.

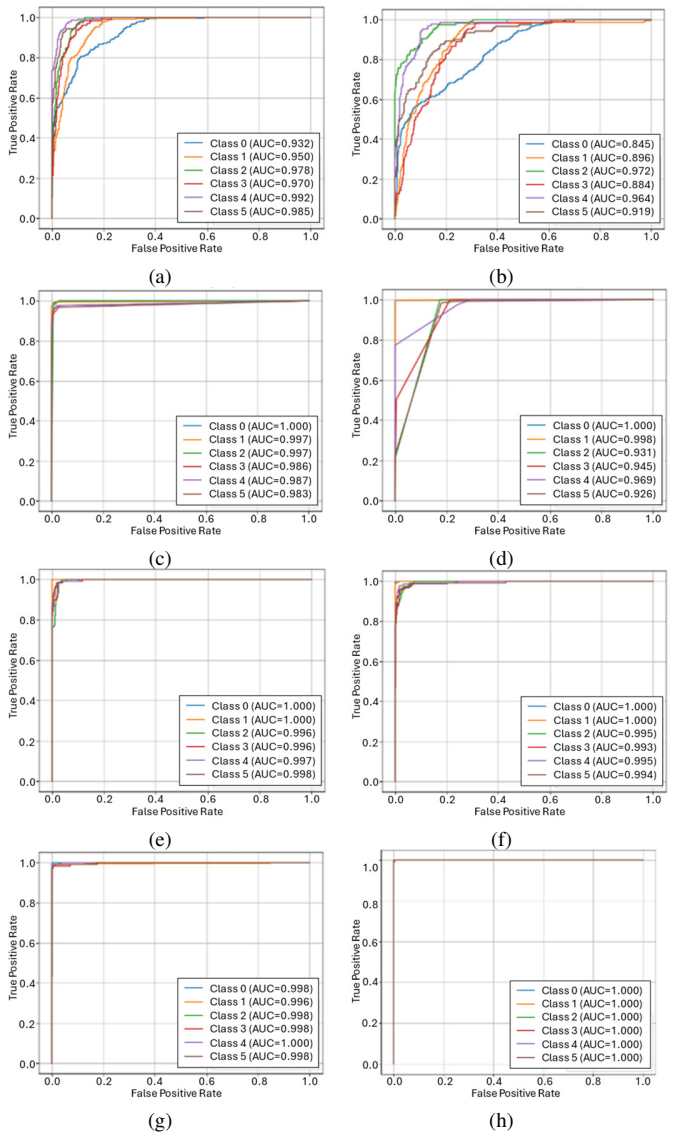

Fig. 3. OvR ROC curves for eight classifiers on a six-class: (a) MLR, (b) GNB, (c) KNN, (d) DT, (e) RF, (f) SVC (RBF), (g) MLP, (h) Adaptive CatBoost.

These patterns fit with the models' capacity and inductive bias. Boosted trees and neural networks may find nonlinear connections between the 18 IMU channels, which gives them AUCs that are almost perfect. RF, KNN, and SVC (RBF) also perform well at modeling nonlinear structure, but there is a little more variability between classes. When classes cannot be separated linearly, MLR does not work as well because it puts linear bounds on the data. GNB's requirements of conditional independence and Gaussianity do not apply to sensor features that are correlated. A single shallow DT does not have the ensemble averaging that keeps performance stable.

When operating points need to change (like when high-recall screening is needed instead of high-precision alerts), the models in the top two tiers offer stable trade-offs across thresholds for all classes. The lower AUCs in MLR, GNB, and the single DT suggest that feature engineering (temporal and spectral statistics), class-specific thresholding, probability calibration, or replacing them with ensemble or neural approaches for mission-critical deployments could all be useful.

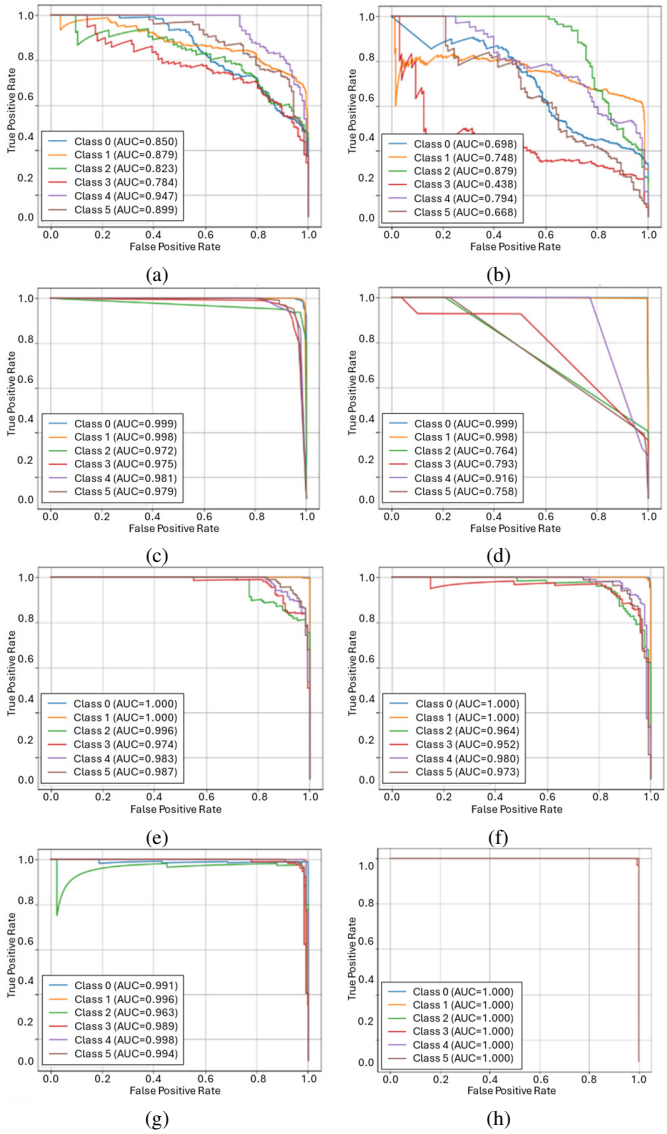

Fig. 4. OvR PR curves for eight classifiers: (a) MLR, (b) GNB, (c) KNN, (d) DT, (e) RF, (f) SVC (RBF), (g) MLP, (h) Adaptive CatBoost.

In Figure 4, a significant separation in performance can be observed. As shown in Figure 4(g, h), across all classes, Adaptive CatBoost and MLP (32–16) get close to the ceiling PR-AUC, with flat, high-precision plateaus over much of the recall range. The middle layer consists of RF (100, depth=6), KNN (k=5), and SVC (RBF), shown in Figure 4 (e, c, f), as their curves are close to the top-right but drop off in precision

more at extreme recall. MLR, GNB, and the single DT (depth=6) show much lower PR-AUC and sharper drops in precision, as shown in Figure 4(a, d, b), which means they are more likely to give false positives when sensitivity is increased.

The way the model behaves is in line with inductive biases. In the 18-dimensional feature space, where the class structure is nonlinear, linear boundaries in MLR have trouble, which causes sloped PR curves. The assumptions of conditional independence and Gaussianity in GNB do not work with correlated IMU channels, which makes the curves jagged, and the precision drops sooner. The single shallow DT does not have the variance reduction of ensembles, hence it shows dramatic degradation after modest recall. On the other hand, RF, KNN, and SVC (RBF) are better at finding nonlinear structures. Boosted trees (CatBoost) and the compact neural network (MLP) learn rich interactions that help with almost flawless precision-recall trade-offs.

Class-wise variability indicates that specific activity pairs persist as somewhat overlapping in feature space, as seen by slight separations among class curves within the same model for the middle-tier approaches. These trends suggest that feature engineering could be useful, such as adding temporal window statistics or frequency-domain descriptors, which can better differentiate borderline classes.

From a practical point of view, PR curves help to choose the right threshold. Models with flat, high-lying PR curves (CatBoost or MLP) are good for systems that need high recall without losing accuracy, like safety monitoring. RF strikes a good balance between computational simplicity and interpretability. On the other hand, MLR, GNB, and single DT may need extra steps, class-specific thresholds, probability calibration, or enriched featuresto reach strict precision goals at high recall.

Table II shows the performance of the classifiers on Accuracy (how correct they are overall), Macro-F1 (the unweighted mean F1 across classes, which is sensitive to per-class balance), MCC (Matthews correlation coefficient, a balanced correlation-style summary), and ROC-AUC (OvR - threshold-independent separability).

TABLE II. OVERALL METRICS BY MODEL

Model	Accuracy	MCC	Macro-F1	ROC AUC
CatBoost (Adaptive)	0.9991	0.9989	0.9987	0.999995
MLP (32-16)	0.9872	0.9838	0.9826	0.9980
RF (100x, depth=6)	0.9609	0.9506	0.9394	0.9980
KNN (k=5)	0.9609	0.9505	0.9486	0.9917
SVC (RBF)	0.9464	0.9322	0.9254	0.9962
DT (depth=6)	0.8391	0.8138	0.7445	0.9614
MLR	0.7702	0.7087	0.7695	0.9677
GNB	0.6077	0.5257	0.5888	0.9135

CatBoost (Adaptive) gets close to the best possible score on all measures (Accuracy = 0.9991, MCC = 0.9989, Macro-F1 = 0.9987, ROC-AUC ≈ 1.0000), meaning that it almost perfectly separates classes and calibrates across thresholds. MLP (32–16) is right behind it (Accuracy = 0.9872, MCC = 0.9838), with ROC-AUC = 0.9980, which shows that it is very easy to separate, just like the confusion matrices and PR/ROC curves show. The middle tier consists of RF (100, depth=6), KNN (k=5), and SVC (RBF). RF and KNN have the same Accuracy (0.9609) and similar MCC (~0.95), but KNN has a slightly higher Macro-F1 (0.9486 vs. 0.9394), which means that its performance is a little more consistent. RF and SVC have higher ROC-AUC (0.9980 and 0.9962, respectively), which means that they distinguish classes without using a threshold. The lower tier includes DT (depth=6), MLR, and GNB. The single DT Accuracy (0.8391) and Macro-F1 (0.7445) are much lower than those of ensemble approaches. This shows how forests and boosting can help reduce variation. MLR (Accuracy = 0.7702) does not do well on this IMU feature space that cannot be separated linearly. GNB, on the other hand, has the worst scores (Accuracy = 0.6077, MCC = 0.5257), due to breaking its independence/Gaussian assumptions.

The ROC-AUC values are consistently high for the top five methods across all models, showing that many classifiers can separate classes well. MCC and Macro-F1, on the other hand, offer sharper discrimination by punishing class-wise imbalances and error structure. CatBoost is the best choice for situations that need the most accuracy and dependability. The small MLP is a good alternative. When prioritizing a balance between accuracy and explainability, RF is an excellent choice. LR and GNB should be kept as baseline benchmarks.

Table III shows the F1-scores for eight classifiers, one for each class (C0-C5). CatBoost (Adaptive) is consistently strong, scoring 1.000 on C0, C1, C2, and C5, and almost perfect on C3 (0.9960) and C4 (0.9961). The MLP (32–16) is also high, but it drops slightly on C2 (0.9721) and C3 (0.9647). RF (100, depth=6) is still great on C0 (0.9984), C1 (1.000), and C4 (0.9647), but dropped a lot on C2 (0.8923), C3 (0.8726), and C5 (0.9083). KNN (k=5) is well balanced across classes (0.9291–0.9850), while SVC (RBF) is a little lower, especially on C2 (0.8706) and C3 (0.8828). The single DT is very different from the others, as it is almost perfect on C0 and C1 but much worse on C2 (0.5754), C3 (0.6531), and C5 (0.3709). MLR has low results, especially on C0 (0.7103) and C3 (0.6867). GNB gets the lowest scores overall, with C3 (0.4533) and C4 (0.5538) being the most affected. C0 and C1 are always the easiest across models, but C2, C3, and C5 are harder for weaker approaches, which suggests that the features are similar across classes. Adding temporal or spectral features and employing nonlinear or ensemble models can help with these class-specific problems.

TABLE III. PER-CLASS F1-SCORE

Model	C0	C1	C2	C3	C4	C5
CatBoost (Adaptive)	1.000	1.000	1.000	0.9960	0.9961	1.000
MLP (32-16)	0.9951	0.9959	0.9721	0.9647	0.9881	0.9796
RF (100x, depth=6)	0.9984	1.000	0.8923	0.8726	0.9647	0.9083
KNN (k=5)	0.9803	0.9850	0.9453	0.9195	0.9291	0.9322
SVC (RBF)	0.9743	0.9905	0.8706	0.8828	0.9206	0.9138
DT(depth=6)	0.9983	0.9973	0.5754	0.6531	0.8722	0.3709
MLR	0.7103	0.8088	0.7491	0.6867	0.8571	0.8051
GNB	0.5777	0.7041	0.6908	0.4533	0.5538	0.5531

*C0) down_stairs; C1) up_stairs; C2) down_slope; C3) up_slope; C4) down_largestair; C5) up_largestair

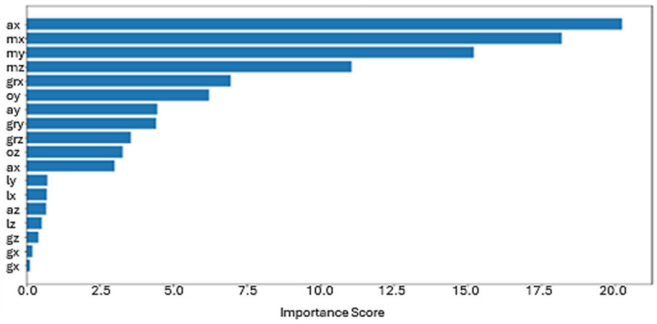

Fig. 5. CatBoost feature importance.

The feature-importance profile in Figure 5 shows that global position and heading are the most important factors. The orientation and magnetometer channels make up the biggest parts: o_x (≈20%) is the most important feature, while the magnetometer triad m_x, m_y, m_z makes up about 44% of the total. This pattern is similar to how a wheelchair that can climb stairs moves: it lines up with the stairs, chooses an up or down trajectory, and then reorients itself on landings. Yaw/heading (o_x: magnetometer-referenced) differentiates approach and turning behaviors, whereas pitch/roll signals (reflected in orientation angles and gravity components) distinguish going up and down and the level posture during entry and exit.

The gravity-vector components (gr_x, gr_y, gr_z: ≈15% combined) are the second most important, as they show the tilt in relation to gravity. While going up, the chair's frame has a constant positive pitch. While going down, the pitch changes, and while going level, gravity lines up differently with the chassis axes. These quasi-static tilt discrepancies are strong indications for a stair-climbing system that is tracked or stabilized. This kind of device tends to smooth out sudden changes in motion while keeping the posture.

In this non-temporal, per-sample form, pose/heading is more important than accelerometer axes (a_x, a_y, a_z: ≈ 8%). Linear acceleration in stair-climbing wheelchairs is frequently controlled and low-jerk, especially when the mechanism meters each step, so instantaneous accelerations provide limited separability without temporal context. Gyroscope rates (g_x, g_y, g_z) and linear acceleration (l_x, l_y, l_z) are the least important. This is normal when the model takes in single samples instead of windowed statistics. Angular velocities and translational bursts become useful when averaged over time (e,g., mean, variance, peak rate, or spectral energy around the tread cadence).

The prevalence of o_x shows that angle wrap-around (0/360°) may create severe thresholds that enhanced trees exploit. Using sin/cos pairs for angles can smooth out circular discontinuities, which can typically stabilize splits and spread importance more evenly across orientation channels. Since magnetometers are so common, it is important to pay attention to magnetic distortions. For example, steel-reinforced interior stairwells might affect headings. Regular hard/soft-iron calibration and validation in the target environment helps keep the features anchored by magnetometers reliable.

Fig. 6. Embedded CatBoost deployment.

Deployment for real-time tests was carried out (Figure 6). The CatBoost model (.pkl) was trained, exported in the native CatBoost format (.cbm), and then converted to a C header file containing the model as a byte array to embed it into the ESP32 firmware. This header file was added to the ESP32 project to load into the microcontroller at runtime. After compiling and flashing the firmware on the ESP32, the device was able to perform real-time inference by reading IMU data, forming the required 18-channel feature vector, and running the CatBoost inference routine for wheelchair activity class prediction. The CatBoost model, exported in the embedded format header, achieved 0.9785 accuracy offline on the combined stair-climbing dataset. In real-time operation on a wheelchair, a slight degradation is expected due to sensor noise, mounting variation, and environmental disturbances, but accuracy should remain high under conditions similar to those in data collection.

IV. CONCLUSIONS

This study demonstrates that window-free, single-sample IMU classification can accurately identify stair-climbing wheelchair actions with near-perfect precision, while remaining comprehensible and computationally efficient. The adaptive CatBoost configuration was the best of the eight classifiers (Accuracy ≈ 0.999, MCC ≈ 0.999). The compact MLP was second, and the strong middle tier consisted of RF, KNN, and SVC (RBF). The weakest classifiers were MLR, GNB, and a single DT. ROC/PR analyses confirmed remarkable threshold-independent separability; however, per-class F1 patterns indicated that more challenging classes, such as C2, C3, and C5, mostly affect weaker models. A feature-importance analysis revealed a hierarchy of posture and heading: the orientation and magnetometer channels are the most significant, the gravity channel is the second most important, and the instantaneous gyroscope and linear acceleration channels are the least relevant when there is no time context. This fits with how things move when they approach, go up, go down, and land. These results support low-latency, on-device deployment and give clear advice on how to pick features for embedded controllers. Future improvements should look into short causal windows or spectral cues to improve dynamic sensors without increasing latency, add angle sine/cosine encodings and robust magnetometer calibration to improve stability, and test cross-device generalization along with model compression/distillation for platforms with limited resources.

ACKNOWLEDGMENT

The authors sincerely acknowledge the Department of Mechatronics Engineering at Rajamangala University of Technology Thanyaburi (RMUTT) for their assistance, resources, and research facilities that enabled this study.

REFERENCES

[1] Y. Zhu, H. Li, S. Lyu, X. Shan, Y. K. Jan, and F. Ma, "Stair-climbing wheelchair proven to maintain user's body stability based on AnyBody musculoskeletal model and finite element analysis," *PLOS ONE*, vol. 18, no. 1, Jan. 2023, Art. no. e0279478, https://doi.org/10.1371/journal.pone.0279478.

[2] R. Baud, A. R. Manzoori, A. Ijspeert, and M. Bouri, "Review of control strategies for lower-limb exoskeletons to assist gait," *Journal of NeuroEngineering and Rehabilitation*, vol. 18, no. 1, Dec. 2021, Art. no. 119, https://doi.org/10.1186/s12984-021-00906-3.

[3] E. Bergamini, G. Ligorio, A. Summa, G. Vannozzi, A. Cappozzo, and A. Sabatini, "Estimating Orientation Using Magnetic and Inertial Sensors and Different Sensor Fusion Approaches: Accuracy Assessment in Manual and Locomotion Tasks," *Sensors*, vol. 14, no. 10, pp. 18625–18649, Oct. 2014, https://doi.org/10.3390/s141018625.

[4] O. D. Lara and M. A. Labrador, "A Survey on Human Activity Recognition using Wearable Sensors," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 3, pp. 1192–1209, 2013, https://doi.org/10.1109/SURV.2012.110112.00192.

[5] O. Banos, J. M. Galvez, M. Damas, H. Pomares, and I. Rojas, "Window Size Impact in Human Activity Recognition," *Sensors*, vol. 14, no. 4, pp. 6474–6499, Apr. 2014, https://doi.org/10.3390/s140406474.

[6] D. T. Dat, L. H. Ky, S. Hoang, and T. D. Thuan, "Building an Anti-Tip System for Robots Transporting People Up and Down the Stairs," *Engineering, Technology & Applied Science Research*, vol. 15, no. 3, pp. 23756–23766, June 2025, https://doi.org/10.48084/etasr.10618.

[7] A. Wang, G. Chen, J. Yang, S. Zhao, and C. Y. Chang, "A Comparative Study on Human Activity Recognition Using Inertial Sensors in a Smartphone," *IEEE Sensors Journal*, vol. 16, no. 11, pp. 4566–4578, June 2016, https://doi.org/10.1109/JSEN.2016.2545708.

[8] K. Chen, D. Zhang, L. Yao, B. Guo, Z. Yu, and Y. Liu, "Deep Learning for Sensor-based Human Activity Recognition: Overview, Challenges, and Opportunities," *ACM Computing Surveys*, vol. 54, no. 4, pp. 1–40, May 2022, https://doi.org/10.1145/3447744.

[9] W. Chen, J. Li, S. Zhu, X. Zhang, Y. Men, and H. Wu, "Gait Recognition for Lower Limb Exoskeletons Based on Interactive Information Fusion," *Applied Bionics and Biomechanics*, vol. 2022, pp. 1–19, Mar. 2022, https://doi.org/10.1155/2022/9933018.

[10] W. H. K. De Vries, R. M. A. Van Der Slikke, M. P. Van Dijk, and U. Arnet, "Real-Life Wheelchair Mobility Metrics from IMUs," *Sensors*, vol. 23, no. 16, Aug. 2023, Art. no. 7174, https://doi.org/10.3390/s23167174.

[11] H. Najeh, C. Lohr, and B. Leduc, "Real-Time Human Activity Recognition on Embedded Equipment: A Comparative Study," *Applied Sciences*, vol. 14, no. 6, Mar. 2024, Art. no. 2377, https://doi.org/10.3390/app14062377.

[12] A. V. Dorogush, V. Ershov, and A. Gulin, "CatBoost: gradient boosting with categorical features support." arXiv, Oct. 24, 2018, https://doi.org/10.48550/arXiv.1810.11363.

[13] W. S. Lima, E. Souto, K. El-Khatib, R. Jalali, and J. Gama, "Human Activity Recognition Using Inertial Sensors in a Smartphone: An Overview," *Sensors*, vol. 19, no. 14, July 2019, Art. no. 3213, https://doi.org/10.3390/s19143213.

[14] G. McClatchey, M. Goršič, M. R. Adelman, W. C. Kephart, and J. R. Rammer, "Holistic Sensor-Based Approach for Assessing Community Mobility and Participation of Manual Wheelchair Users in the Real World," *Journal of Sensor and Actuator Networks*, vol. 13, no. 6, Oct. 2024, Art. no. 70, https://doi.org/10.3390/jsan13060070.

[15] J. Wang, Y. Chen, S. Hao, X. Peng, and L. Hu, "Deep learning for sensor-based activity recognition: A survey," *Pattern Recognition Letters*, vol. 119, pp. 3–11, Mar. 2019, https://doi.org/10.1016/j.patrec.2018.02.010.

[16] D. Su, Z. Hu, J. Wu, P. Shang, and Z. Luo, "Review of adaptive control for stroke lower limb exoskeleton rehabilitation robot based on motion intention recognition," *Frontiers in Neurorobotics*, vol. 17, July 2023, Art. no. 1186175, https://doi.org/10.3389/fnbot.2023.1186175.

[17] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, Oct. 2001, https://doi.org/10.1023/A:1010933404324.

[18] D. Chicco and G. Jurman, "The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation," *BMC Genomics*, vol. 21, no. 1, Dec. 2020, Art. no. 6, https://doi.org/10.1186/s12864-019-6413-7.

[19] Y. Tian, J. Zhang, J. Wang, Y. Geng, and X. Wang, "Robust human activity recognition using single accelerometer via wavelet energy spectrum features and ensemble feature selection," *Systems Science & Control Engineering*, vol. 8, no. 1, pp. 83–96, Jan. 2020, https://doi.org/10.1080/21642583.2020.1723142.

[20] S. Mekruksavanich and A. Jitpattanakul, "Deep Convolutional Neural Network with RNNs for Complex Activity Recognition Using Wrist-Worn Wearable Sensor Data," *Electronics*, vol. 10, no. 14, July 2021, Art. no. 1685, https://doi.org/10.3390/electronics10141685.

[21] F. Bo, J. Li, W. Wang, and K. Zhou, "Robust Attitude and Heading Estimation under Dynamic Motion and Magnetic Disturbance," *Micromachines*, vol. 14, no. 5, May 2023, Art. no. 1070, https://doi.org/10.3390/mi14051070.

[22] P. Chawaphan, D. Maneetham, and P. N. Crisnapati, "Stairclimbing Wheelchair Dataset," *Mendeley Data*, vol. 1, Dec. 2024, https://doi.org/10.17632/jchzh3zwcy.1.

[23] A. R. Rachakonda and A. Bhatnagar, "A : Extending area under the ROC curve for probabilistic labels," *Pattern Recognition Letters*, vol. 150, pp. 265–271, Oct. 2021, https://doi.org/10.1016/j.patrec.2021.06.023.

[24] Q. Lin, G. Ye, J. Wang, and H. Liu, "RoboFlow: a Data-centric Workflow Management System for Developing AI-enhanced Robots," in *Proceedings of the 5th Conference on Robot Learning*, Jan. 2022, pp. 1789–1794.

[25] A. De La Cruz Huayanay, J. L. Bazán, and C. M. Russo, "Performance of evaluation metrics for classification in imbalanced data," *Computational Statistics*, Aug. 2024, https://doi.org/10.1007/s00180-024-01539-5.

[26] D. Chicco and G. Jurman, "The Matthews correlation coefficient (MCC) should replace the ROC AUC as the standard metric for assessing binary classification," *BioData Mining*, vol. 16, no. 1, Feb. 2023, Art. no. 4, https://doi.org/10.1186/s13040-023-00322-4.

[27] E. Christodoulou, J. Ma, G. S. Collins, E. W. Steyerberg, J. Y. Verbakel, and B. Van Calster, "A systematic review shows no performance benefit of machine learning over logistic regression for clinical prediction models," *Journal of Clinical Epidemiology*, vol. 110, pp. 12–22, June 2019, https://doi.org/10.1016/j.jclinepi.2019.02.004.

[28] A. Ali, W. Samara, D. Alhaddad, A. Ware, and O. A. Saraereh, "Human Activity and Motion Pattern Recognition within Indoor Environment Using Convolutional Neural Networks Clustering and Naive Bayes Classification Algorithms," *Sensors*, vol. 22, no. 3, Jan. 2022, Art. no. 1016, https://doi.org/10.3390/s22031016.

[29] A. N. Beskopylny *et al.*, "Concrete Strength Prediction Using Machine Learning Methods CatBoost, k-Nearest Neighbors, Support Vector Regression," *Applied Sciences*, vol. 12, no. 21, Oct. 2022, Art. no. 10864, https://doi.org/10.3390/app122110864.

[30] I. D. Mienye and N. Jere, "A Survey of Decision Trees: Concepts, Algorithms, and Applications," *IEEE Access*, vol. 12, pp. 86716–86727, 2024, https://doi.org/10.1109/ACCESS.2024.3416838.

[31] Y. Wang and Y. Li, "Mapping the ratoon rice suitability region in China using random forest and recursive feature elimination modeling," *Field Crops Research*, vol. 301, Oct. 2023, Art. no. 109016, https://doi.org/10.1016/j.fcr.2023.109016.

[32] S. Y. Sheikh and M. T. Jilani, "A ubiquitous wheelchair fall detection system using low-cost embedded inertial sensors and unsupervised one-class SVM," *Journal of Ambient Intelligence and Humanized Computing*, vol. 14, no. 1, pp. 147–162, Jan. 2023, https://doi.org/10.1007/s12652-021-03279-6.

[33] L. Madaoui, O. Kerdjidj, and M. Kedir-Talha, "Design and implementation of IMU-based locomotion mode recognition system on Zynq SoC," *Microprocessors and Microsystems*, vol. 102, Oct. 2023, Art. no. 104927, https://doi.org/10.1016/j.micpro.2023.104927.

[34] A. K. Sharma, S. H. Liu, X. Zhu, and W. Chen, "Predicting Gait Parameters of Leg Movement with sEMG and Accelerometer Using

CatBoost Machine Learning," *Electronics*, vol. 13, no. 9, May 2024, Art. no. 1791, https://doi.org/10.3390/electronics13091791.