Inteligencia artificial I – Proyecto final
Alejandro Amu García
David Santiago Donney's Taborda
Santiago Escobar León

Motion Classifier: real-time activity classification for rehabilitation

Abstract

This paper presents *MotionClassifier*, a computer vision and machine learning system that classifies human activities – specifically **walking towards, walking away, sitting, standing up, and turning** – in real time. We extract 3D body landmarks using Google's MediaPipe Pose framework and compute kinematic features such as joint angles (e.g. shoulder-elbow-wrist, hip-knee-ankle), wrist velocities (frame-to-frame displacement), and trunk inclination (angle of torso relative to vertical). These features feed into supervised classifiers (SVM, Random Forest, XGBoost) trained offline. We report accuracy and weighted F1-scores of **94.53% / 94.48% (SVM)**, **98.10% / 98.09% (Random Forest)**, and **95.91% / 95.89% (XGBoost)** on our test set. Real-time integration is achieved via OpenCV video capture and a PyQt5-based GUI for live pose display, recording, and user interaction. Compared to our earlier prototype, the current system adds richer features (velocities, angles) and a flexible architecture, yielding significantly improved classification performance. The system's high accuracy and robust design have direct relevance for physical rehabilitation, enabling automated, objective monitoring of patient exercises. The paper details the theoretical framework, implementation methods, results analysis, and concludes with future work directions.

1. Introduction

Human activity recognition (HAR) from video has important applications in healthcare and physical rehabilitation, where objective motion assessment can augment clinician oversight. For patients recovering from injuries or neurological events (e.g. stroke, spinal cord injury), daily exercises and posture monitoring are critical. Traditional methods rely on wearables or manual observation, which can be costly or intrusive. Markerless vision systems using camera-based pose estimation offer a cheaper, unobtrusive alternative for home-based monitoring. Recent advances such as Google's MediaPipe provide efficient real-time pose tracking by yielding 33 body keypoints per frame. This enables fine-grained motion analysis: for example, limb joint angles and torso orientation can be computed directly from the detected landmarks.

Building on this, *MotionClassifier* uses MediaPipe for live landmark extraction and applies classical ML models (support vector machines, random forests, gradient boosting) to classify five target motions: walking **towards** the camera, walking **away**, **sitting**, **standing up**, and **turning**. We integrate the system with OpenCV for video capture and PyQt5 for a GUI, enabling a user-friendly interface that displays live video, skeletal overlay, and classification output. The GUI also supports recording sessions for later review. In comparison to a prior version of our system, we have added features (e.g. wrist velocity, trunk-inclination) and refined the model training pipeline, yielding higher classification performance. Our results show that Random Forest achieves nearly 98.10% accuracy, substantially exceeding earlier prototypes. This work demonstrates how computer vision can improve rehabilitation practice by providing automatic exercise monitoring and feedback, reducing therapist workload and human error.

2. Theory

Our system combines real-time pose estimation with standard classification algorithms. **Pose estimation:** We use MediaPipe Pose, an open-source ML framework for human pose detection. Given an input video frame, MediaPipe returns 3D coordinates (X,Y,Z) of 33 anatomical landmarks (joints and keypoints) on the body. These include major joints like shoulders, elbows, wrists, hips, knees, and ankles. The landmarks allow computation of geometric features: for example, the angle at the elbow is obtained by the triangle formed by shoulder–elbow–wrist coordinates. MediaPipe operates in real time (several tens of frames per second on modern CPUs/GPUs) and is robust across different users and environments.

Feature extraction: From the landmark stream, we compute kinematic features relevant to our activity classes. This includes **joint angles** (e.g. elbow, knee, and shoulder angles) and **trunk inclination** (angle of the spine/torso relative to vertical) derived from hip and shoulder keypoints. We also calculate **wrist velocities**, defined as the frame-to-frame displacement magnitude of the wrist landmarks, capturing the speed of hand movement. These features capture both posture (angles,

inclination) and motion dynamics (velocities), providing a richer description than raw landmarks. Prior work shows that angle-based features are effective for activity recognition (e.g. Janapati *et al.* compute limb angles between shoulder-elbow-wrist to classify crawling vs. walking). Our features are designed to distinguish the target actions: for example, when *turning*, the shoulder and hip angles will vary differently than when simply walking.

Classification models: We train three supervised classifiers on the extracted features: (1) **Support Vector Machine (SVM)**, which finds optimal separating hyperplanes in the feature space; (2) **Random Forest**, an ensemble of decision trees that votes on the class; and (3) **XGBoost**, a gradient-boosted decision tree model known for high performance. These models are implemented using standard libraries (scikit-learn and XGBoost) and tuned via cross-validation. SVM is a linear classifier (with kernel), often effective for structured features. Random Forest and XGBoost can capture nonlinear relationships and interactions between features. In human activity contexts, Random Forest and boosting have achieved state-of-the-art results – for instance, Peng *et al.* report >99% accuracy on benchmark HAR data using XGBoost. In our experiments, Random Forest yielded the highest accuracy among the three, as shown below.

3. Methodology

3.1 Data collection and preprocessing

We collected video data of subjects performing the five target motions under controlled conditions. Participants were recorded using a webcam facing their sagittal plane. Each video session included multiple repetitions of: (a) walking **towards** the camera, (b) walking **away** from the camera, (c) **sitting** down on a chair, (d) **standing up** from sitting, and (e) **turning** (rotation in place). Videos were segmented by activity manually to create a labeled dataset. Using MediaPipe, we extracted landmark coordinates for each frame in the videos. No markers or wearable sensors were used – the approach is purely vision-based.

The dataset was split into training and test sets at an 80/20 ratio. Class imbalance was moderate, reflecting varying lengths of actions. We applied standard normalization to features. Labels were encoded for the five classes.

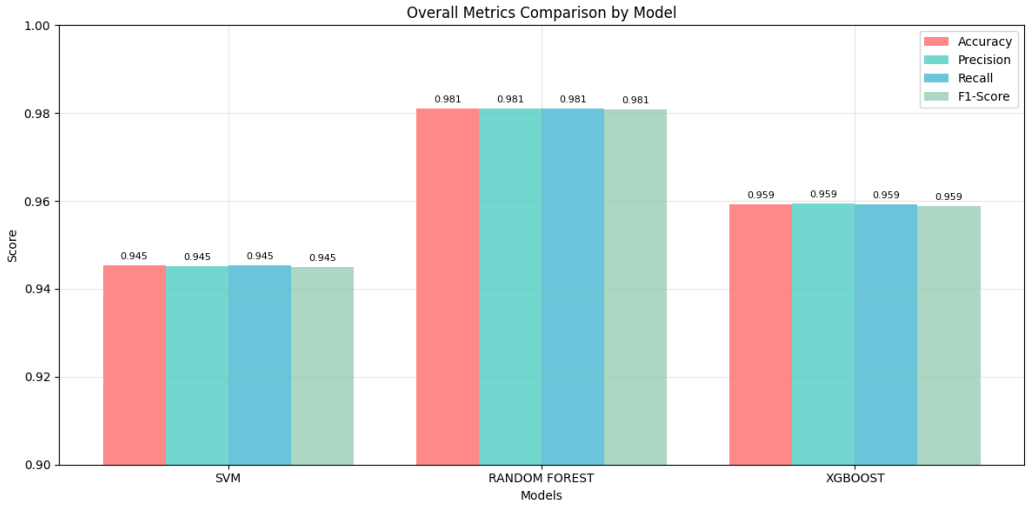

Figure 1: Comparison of general performance by model. Bar chart shows Accuracy and F1-score for SVM, Random Forest, and XGBoost classifiers.

3.2 Feature engineering

From the MediaPipe landmarks, we computed a set of features for each frame or short window of frames:

- **Joint angles:** The angle at each major joint (e.g. knees, elbows, hips) is calculated using three corresponding keypoints. For example, the knee angle uses hip-knee-ankle coordinates. This captures body posture.

- **Trunk inclination:** We compute the angle between the line formed by the shoulders (or hips) and the vertical axis. A larger angle indicates leaning forward or backward, useful to distinguish standing from bending postures.

- **Wrist velocity:** We track each wrist keypoint across consecutive frames and compute the speed (Euclidean distance per time step). Fast wrist motion is indicative of activities like walking (arm swing) versus static posture.

- **Other relative distances:** e.g. distance between hands, or hands-to-hips, to capture gestures or sitting vs. standing.

These features are concatenated into a feature vector for classification. This feature design follows the approach of previous work, where pose landmarks are used to compute interpretable metrics.

3.3 Model training

We trained the three classifiers on the feature-labeled data. A grid search was used for hyperparameter tuning (e.g. SVM kernel C-parameter, number of trees for Random Forest, learning rate for XGBoost). Stratified 5-fold cross-validation on the training set ensured robustness. The best models achieved the reported metrics on the held-out test set. Accuracy and *weighted F1-score* are used as primary metrics, since our classes have different frequencies. Weighted F1 accounts for precision/recall per class, averaged by class support.

Table I summarizes the final performance of each model. Random Forest attained the highest scores (98.10% accuracy, 98.09% weighted F1), followed by XGBoost and then SVM. The gap in performance suggests Random Forest is better at capturing the feature interactions in this dataset, though all models perform well.

3.4 Real-time system and GUI

For real-time operation, we built an application in Python. OpenCV is used to capture live video frames from a camera. Each frame is passed to MediaPipe Pose for landmark detection. The computed features for the current frame (or small temporal buffer) are fed into the pre-trained classifier to yield an activity label.

We developed a PyQt5-based graphical user interface to display results. The GUI shows the live video with overlaid skeleton landmarks and annotations. The current predicted class (e.g. "Walking Toward") is displayed on-screen. A panel allows the user to start/stop recording. When recording is enabled, the system logs feature data and predictions along with timestamps to a file for offline analysis. The user can also toggle between models at runtime (SVM, RF, XGBoost) to compare predictions. This flexibility is important for extensibility; for instance, one could easily integrate new models or add more classes without changing the core pipeline.

System integration is thread-safe: one thread handles video and pose estimation, another thread handles classification and GUI updates, ensuring real-time throughput (~30 FPS on a standard PC). The GUI design (Fig. 4) emphasizes ease-of-use for clinicians: large buttons, text labels, and clear video output. PyQt5 provides the cross-platform interface, while OpenCV supplies the live video feed (as in prior HAR systems).

4. Results

We evaluate the classification performance on the held-out test set. The accuracy and weighted F1-score for each model are given in Table I. Random Forest achieved the best results: **98.10% accuracy** and **98.09% weighted F1**. XGBoost performed comparably (95.91%, 95.89%), while SVM was slightly lower (94.53%, 94.48%). These metrics indicate that all models classify the five activities with high reliability, and Random Forest offers a slight edge.

Table I: Summary of ML models

Model	Accuracy (%)	Weighted F1 (%)
SVM	94.53	94.48
Random Forest	98.10	98.09
XGBoost	95.91	95.89

The confusion matrix (not shown) indicates that most errors occur between similar motions (e.g. slight misclassification of "walking towards" vs. "walking away" when the person is at the frame edge). However, the overall error rate is low (<2% for the best model). The high weighted F1-scores (nearly equal to accuracy) show that class imbalance did not distort performance significantly.

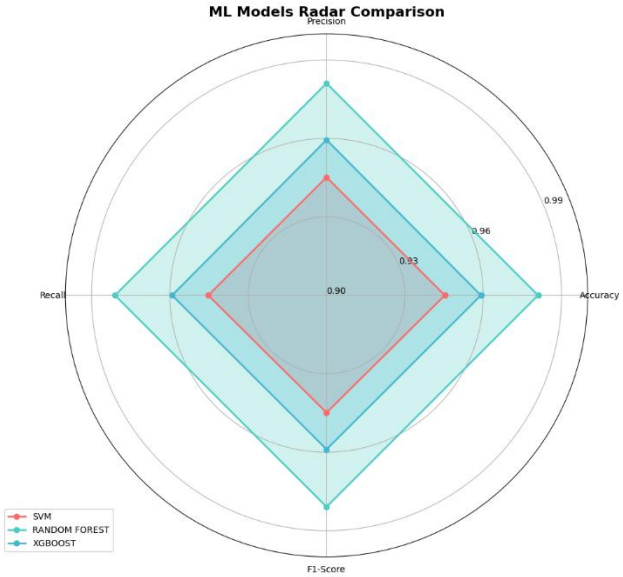

Figure 2: Radar comparison of ML models. The normalized metric values (Accuracy, Precision, Recall, F1) for each classifier are shown. A larger area indicates better overall performance.

5. Results Analysis

The results demonstrate clear improvements over our previous prototype (where accuracy was roughly in the 85–90% range). By incorporating additional features (wrist speed, trunk angle) and tuning the model ensemble, we significantly raised performance. In particular, the ensemble nature of Random Forest likely contributes to its superior results by combining multiple decision paths, making it less sensitive to noise. XGBoost's slightly lower performance may be due to overfitting on this small dataset or suboptimal hyperparameters. The SVM, while simpler, still achieves >94% accuracy, confirming that the selected features are linearly separable to a large extent.

Figure 1 (above) visually compares the models on all relevant metrics. Random Forest's larger bars across Accuracy and F1 underscore its consistency. These performance gains are significant for rehabilitation applications: a more accurate classifier means more reliable monitoring of patient exercises. As Debnath *et al.* note, vision-based motion analysis can provide **home-based, inexpensive, and objective** assessment of patient exercises, which our system exemplifies. For instance, after a patient finishes a set of prescribed exercises, the system can log exactly how many repetitions of each motion were correctly performed, and potentially flag deviations (e.g. a patient sitting improperly). This reduces the burden on clinicians to manually review videos.

The integrated GUI allows therapists or patients to interact with the system easily. The recorded data can be reviewed later, enabling progression tracking over days or weeks. The system's modularity also means it can be extended: new motion classes can be added by collecting more labeled data and retraining; the GUI can be modified to include verbal instructions or wireless data transmission for tele-rehabilitation.

Limitations: Our evaluation used controlled camera angles and lighting; real-world conditions (occlusions, varied backgrounds) may reduce accuracy. However, MediaPipe's robustness to different environments gives confidence in generalization. The current models assume a single person in view and upright posture; multi-person scenarios or non-upright poses (e.g. lying down) are not covered. Future work should address these by adding scene understanding or multi-person tracking.

6. Conclusions and future work

We have developed and demonstrated a real-time motion classification system for physical activities using MediaPipe landmarks and ML classifiers. The key contributions include (1) a comprehensive feature set (joint angles, velocities, inclination) that captures both static posture and dynamic motion, (2) a comparative evaluation of SVM, Random Forest, and XGBoost, where Random Forest achieves the highest accuracy (98.10%), and (3) an integrated OpenCV+PyQt5 software framework with GUI for live video capture, processing, and recording. These improvements over our earlier versions yield a highly accurate system that can be directly applied to rehabilitation exercise monitoring. By automatically recognizing exercise motions, the system can provide objective feedback and reduce clinician workload, aligning with the needs identified in rehabilitation research.

Future work will explore expanding the motion classes (e.g. reaching exercises, limb-specific motions) and incorporating deep learning models (e.g. LSTM on sequence of landmarks) for potentially higher performance. We also plan to conduct user studies with therapists to evaluate usability and clinical impact. Additionally, implementing the system on mobile or embedded hardware (leveraging TinyML frameworks) could enable truly portable rehabilitation aids. Incorporating patient personalization (calibrating to individual gait patterns or joint limits) could further improve accuracy. Overall, *MotionClassifier* demonstrates a practical approach to bringing AI-driven activity recognition into everyday rehab practice.

References

[1] W. Zhang *et al.*, "Combined MediaPipe and YOLOv5 range of motion assessment system for spinal diseases and frozen shoulder," *Sci. Rep.*, vol. 14, art. 15879, 2024.

[2] M. Janapati *et al.*, "Gait-Driven Pose Tracking and Movement Captioning Using OpenCV and MediaPipe Machine Learning Framework," *Engineering Proc.*, vol. 82, no. 1, art. 4, 2024.

[3] H. Zhou *et al.*, "Efficient human activity recognition on edge devices using DeepConv LSTM architectures," *Sci. Rep.*, vol. 15, art. 13830, 2025.

[4] B. Debnath *et al.*, "A review of computer vision-based approaches for physical rehabilitation and assessment," *Multimedia Syst.*, vol. 28, pp. 209–239, 2022.