Final Report: Development of a Human Activity Recognition and Real-Time Joint Tracking System

Faculty of Engineering, Design and Applied Sciences Department of Computing and Intelligent Systems Systems Engineering

Teachers: Uram Sosa Aguirre, Milton Sarria Paja
Subject: Artificial Intelligence I
Institution: Icesi University

Members:

Daniel Esteban Jaraba Gaviria, Santiago Angel Ordoñez, Santiago Gutiérrez Villegas

Delivery Day:

June 13, 2025

Index:

Summary

This project addresses the development of a software system for real-time human activity recognition (HAR) and postural analysis. Using the CRISP-DM methodology, a complete solution was designed and implemented. It captures video through a camera, processes the frames to extract a skeleton of 18 key joints using the MediaPipe library, and classifies the activity executed by a person into one of five predefined categories: walking towards the camera, walking back, turning, sitting down, and standing up. The core of the system is a supervised learning model, specifically a Support Vector Machine (SVM), trained with biomechanical features such as joint angles, velocities, and trunk inclination. The final system offers a live visualization that overlays the detected skeleton, the classified activity with its confidence level, and the frames per second (FPS) of the processing, demonstrating the feasibility of applying computer vision and machine learning techniques for low-latency motion analysis.

Introduction

Human Activity Recognition (HAR) is a prominent field of research within artificial intelligence and computer vision, with applications ranging from medical rehabilitation and sports analysis to human-robot interaction and intelligent surveillance systems. A system's ability to understand and quantify human movement in real time opens up a range of possibilities for creating more intuitive and contextual support tools.

Problem Description The objective of this project is to develop a software tool capable of analyzing and classifying five basic human activities (walking toward the camera, walking back, turning, sitting, and standing) from a real-time video source. In addition to classification, the system must track joint movements and key postural parameters, such as trunk tilt and knee angles. The main technical challenge lies in processing the sequence of poses, extracted frame by frame, to perform supervised classification and time-series analysis under strict low-latency constraints that enable a fluid and interactive response.

Relevance and Context The importance of a system like this lies in its potential to provide immediate and accessible biomechanical feedback. In physical therapy settings, it could guide patients in the correct execution of their exercises; in ergonomics, it could alert them to unhealthy postures in an office setting; and in sports, it could help athletes optimize their technique.

Ethical Considerations: Since its inception, the project has been guided by ethical principles to ensure responsible data handling. Measures implemented include obtaining informed consent from all recorded participants, protecting privacy by not capturing faces or identifiable information, restricting the use of video material exclusively for academic course purposes, promoting diversity to avoid bias in the model, and securely storing the dataset.

Theoretical Foundations

To understand project development, it is necessary to know the following key concepts:

- **Human Pose Estimation:**It is a computer vision technique that detects and locates key joints (landmarks) of a person's body in images or videos. For this project, we used MediaPipe Pose, a Google solution that offers robust, low-latency tracking of 33 3D body landmarks, ideal for real-time applications. Our system focuses on a subset of 18 of these joints to optimize processing.
- **Feature Engineering:**The raw coordinates (x, y, z) of joints, although informative, are not sufficient for a model to distinguish complex activities. Therefore, feature engineering is

used to create more discriminative attributes. In this project, the following were calculated:
    - Joint angles: Such as knee angle and torso tilt, to capture body posture.
    - Velocities and accelerations: Of the wrists and ankles to describe the dynamics of movement.
- **CRISP-DM Methodology:**It is an industry standard for managing data mining and analytics projects. The project adhered to its six phases (business understanding, data understanding, data preparation, modeling, evaluation, and deployment) to ensure structured, iterative, and goal-oriented development.
- **Supervised Classification Models:**Algorithms that learn from pre-labeled data are used. Although several models, such as Random Forest and XGBoost, were considered, the final implementation uses a Support Vector Machine (SVM), a powerful and efficient classifier that searches for the optimal hyperplane separating classes in feature space.

Methodology

The project was executed following the phases of the CRISP-DM methodology, adapted to the specific requirements of video motion analysis.

Understanding Business and Data

In this initial phase, the system objectives and requirements were defined. The problem was framed as a time-series classification challenge with real-time constraints. The five activities to be recognized were established, and the collection of a diverse dataset was planned, including proprietary recordings with multiple volunteers, camera angles, and execution speeds to ensure model generalization.

Data Preparation and Feature Engineering

This was a crucial step implemented through the data_processing.py and feature_engineering.py scripts. The workflow was as follows:

1. **Landmark Extraction:**Input videos (.mp4) were processed to extract, frame by frame, the 3D coordinates (x, y, z) of 18 key joints using MediaPipe Pose. This data, along with the activity label and frame number, was stored in a raw_landmarks.csv file.
2. **Calculation of Derived Characteristics:**From the raw landmarks, biomechanical features such as knee angle, torso tilt, and limb velocity were calculated. These new features were then added to our dataset to enrich it.
3. **Cleaning and Normalization:**The data were cleaned by filling in missing values. StandardScaler was then applied to normalize the features, ensuring that the model was not biased by differences in coordinate scale or the subjects' physical build.
4. **Dimensionality Reduction:**Principal Component Analysis (PCA) was used to reduce the dimensionality of the feature vector, preserving 95% of the variance. This helps speed up training and real-time inference, while also mitigating the risk of overfitting.

Modeling

This phase, implemented in the train_model.py script, focused on training the classifier:

- **Data Division:**The processed dataset was divided into training (80%) and test (20%) sets, using class stratification to handle any imbalance in the number of examples per activity.
- **Classifier Training:**A Support Vector Machine (SVC) model was trained with a Gaussian kernel (RBF) and the parameter probability=True activated, which is necessary to obtain a confidence score in the predictions.
- **Artifact Storage:**The trained SVM model, the label encoder (LabelEncoder), the normalizer (StandardScaler), and the PCA model were saved as .pkl files for later use in the deployment phase.

Assessment

To measure model performance, a set of key metrics were defined that combine classification accuracy with system efficiency:

- Classification Metrics: Precision, Recall and F1-Score to evaluate the balance between false positives and negatives for each class.
- Performance Metrics: Latency (rendering time per frame) and FPS (frames per second) to verify that the solution meets real-time requirements.

Formal evaluation is performed on the test set, which contains data not seen by the model during training.

Deployment

The final phase was to integrate the components into a functional real-time application, as shown in realtime_classifier.py:

1. **Real-Time Logic:**An ActivityClassifier class was developed that loads the serialized artifacts (SVM model, PCA, Scaler and LabelEncoder).
2. **Live Video Processing:**The application captures video from the camera and, for each frame, applies the complete sequence: landmark extraction, feature vector construction and transformation, and prediction.
3. **Heuristics and Prediction Smoothing:**Two key stability improvements were implemented:
    a. **Inactivity Detection:**A total motion threshold (total_movement) allows the user to be classified as "still" heuristically, without overloading the SVM model.
    b. **Temporal Smoothing:**A buffer (a deque of size 10) is used to store the probabilities of the latest predictions. The final classification is based on a weighted average of this buffer, giving more weight to recent frames. This prevents abrupt jumps in classification.
4. **Graphical Interface (GUI):**Using OpenCV, a window was created showing the video with the skeleton overlaid, the label of the predicted activity, the model confidence, a pose analysis with angles, and an FPS counter to monitor performance.

Results

The quantitative results of the SVM model were obtained after evaluation on the test set, which contained 893 samples not seen during training.

Classifier Performance Metrics

The model achieved an overall accuracy of 94%. Detailed performance for each of the eight classes is presented in the following table:

Activity	Precision	Recall (Sensitivity)	F1-Score	Support (Samples)
forward	0.9	0.82	0.86	148
attracted	0.98	0.96	0.97	102
right	1	0.99	0.99	92
lean	0.95	0.93	0.94	88
left	1	0.99	0.99	84
stop	0.97	0.98	0.98	119
back	0.81	0.93	0.87	146

feel	0.99	0.97	0.98	114
Macro Average	0.95	0.95	0.95	893
Weighted Average	0.94	0.94	0.94	893

Source: Classification report generated by *train_model.py.*

Confusion Matrix

Although the graphical display of the confusion matrix is not included, the classification report above allows us to infer the areas of greatest and least success of the model, as detailed in the analysis section.

Real-Time Performance

Qualitative validation was performed through a live demonstration of the system. The application was found to be able to classify a user's actions in front of the camera with low latency, enabling fluid interaction. The FPS counter integrated into the graphical interface confirmed adequate performance for real-time operation on the test hardware.

Analysis of Results

Analysis of the metrics reveals a high-performance model, but with clear areas of opportunity.

- Solid Overall Performance: An accuracy of 94% and an average F1 score (macro and weighted) of 0.95 and 0.94, respectively, indicate that the model is robust and well-balanced overall. The high correlation between the macro and weighted averages suggests that the model is not biased toward classes with more samples.
- High Performance Classes: Activities such as right, left, stand, sit, and back show an F1 score of 0.97 or higher. This demonstrates that engineering characteristics (angles, distances, and orientation) are highly discriminatory for these actions, which involve well-defined postural changes and lateral movements.
- Areas of Confusion and Improvement: The lowest performing classes are moving forward and backward.
    - forward (F1-Score: 0.86): Its recall of 0.82 indicates that the model did not detect 18% of the times when a person was walking forward, probably confusing it with another similar action such as going backward.
    - back off (F1-Score: 0.87): Its accuracy of 0.81 means that of all the times the model predicted "back off", 19% were actually another action.
    - Hypothesis: The confusion between moving forward and backward is logical, as both are forms of walking. The model's difficulty distinguishing them suggests that single-frame pose-based features don't fully capture depth dynamics. Although PCA was key to generalization, it could be filtering out subtle variations in the Z-axis that distinguish these two movements.

Conclusions and Future Work

Conclusions

In this project, a robust human activity recognition system was successfully developed, achieving 94% accuracy using PCA and an SVM classifier. The process taught us the critical importance of detailed feature engineering, the positive impact of dimensionality reduction in preventing overfitting, and the need for rigorous metrics-based evaluation. The implementation of heuristics such as temporal smoothing and inactivity detection proved critical for stability in a real-time application.

This project successfully demonstrated the feasibility of creating a low-cost, noninvasive tool for real-time human movement analysis. The 94% accuracy achieved is not just a technical metric; it translates into a high degree of reliability that could enable practical applications in controlled environments, such as guiding physical therapy patients in the correct execution of their exercises or monitoring basic activities of older adults to detect abnormal mobility patterns.

The key learning from an application perspective is that while static postures and postural changes (such as sitting or standing) are identified with near-perfect accuracy, complex dynamic movements (the confusion between walking forward and walking backward) require more advanced temporal analysis. This distinction is crucial for future development: a system for knee rehabilitation might be nearly ready for piloting, while an application for gait analysis in athletes would need to incorporate sequential models.

In short, more than a technical exercise, this work lays the groundwork for the development of accessible biomechanical feedback systems capable of offering tangible value in areas such as health, wellness, and sports, democratizing access to motion analysis technologies that traditionally require specialized and expensive equipment.

Future Work

To improve and expand the system, the following lines of work are proposed:

- Improving Directional Discrimination: Research and design features that better capture the dynamics of motion along the Z axis to address the confusion between "walking forward" and "walking backward."
- Incorporating Temporal Analysis with Deep Learning: Expand the dataset and experiment with recurrent neural network (LSTM) or Transformer models, which are designed to analyze sequences and could improve accuracy in complex actions.
- Deployment on Edge Devices: Optimize the model (e.g. using quantization) for execution on devices with lower computational power, such as a Raspberry Pi or Jetson Nano, expanding its potential applications.
- Enrich the Interface: Improve the GUI to display historical graphs of angles and velocities, providing more complete biomechanical feedback to the user.

Bibliographic References

1. Google. (n.d.). MediaPipe Pose. Google for Developers. Retrieved from https://ai.google.dev/edge/mediapipe/solutions/vision/pose_landmarker
2. Chapman, P., Clinton, J., Kerber, R., Khabaza, T., Reinartz, T., Shearer, C., & Wirth, R. (2000). CRISP-DM 1.0: Step-by-step data mining guide. SPSS Inc.