*Journal on Interactive Systems, 2026, 17:1*
ISSN: 2763-7719 • doi: 10.5753/jis.2026.6417


**Research Paper**

# Web-Based Application for Assessment of Physical Exercises using Machine Learning

**Rafael Santos** [ SENAI CIMATEC University | *rafael.jesus@aln.senaicimatec.edu.br* ]
**Andrei Sant'anna** [ SENAI CIMATEC University | *andrei.sant'anna@aln.senaicimatec.edu.br* ]
**Luan Machado** [ SENAI CIMATEC University | *luan.vidal@aln.senaicimatec.edu.br* ]
**Lucas Santos** [ SENAI CIMATEC University | *lucas.s.santos@aln.senaicimatec.edu.br* ]
**Márcio Soussa** [ SENAI CIMATEC University | *marcio.soussa@doc.senaicimatec.edu.br* ]

*Av. Orlando Gomes, 1845, Piatã, Salvador, BA, 41650-010, Brazil.*

**Abstract.** The rise of digital platforms for physical activity in the post-pandemic period has driven the demand for accessible and automated solutions to support and correct exercise performance remotely. Recent advances in computer vision have enabled the development of systems that provide real-time feedback using only conventional cameras, eliminating the need for specialized sensors. Using the Design Science Research (DSR) methodology, a web-based solution for real-time exercise assessment powered by machine learning was developed as an artifact. This work covers the entire development pipeline, including data collection and labeling, training of machine learning models, conversion and optimization for inference in the browser environment, and integration into a fully functional and responsive prototype web application. Furthermore, a modular software design is proposed to support scalability and extensibility, allowing the seamless inclusion of new validation strategies. Results show that all trained models achieved accuracies above 90% and executed efficiently in the browser, with inference times below 5 ms across different devices.

**Keywords:** Computer Vision, Pose Estimation, Machine Learning, Physical Exercise, Web Application

**Edited by:** Tadeu Moreira de Classe | **Received:** 22 August 2025 • **Accepted:** 27 February 2026 • **Published:** 12 March 2026

## 1 Introduction

During the COVID-19 pandemic, social distancing measures led to a significant increase in at-home physical activity. The use of digital platforms to support exercise was associated with higher levels of physical activity during this period. Participants who engaged with such platforms were more likely to remain physically active than their inactive peers throughout the study period [Rocha *et al*., 2024].

Despite this growth in digital fitness engagement, the absence of professional supervision and individualized guidance introduced significant health risks. Martinez *et al*. [2020] conducted a large-scale survey on Twitter during the lockdown period and found that 12% of respondents had suffered lockdown-related musculoskeletal injuries, with 5% requiring surgical intervention. Notably, 61% of participants exercised independently, often following non-personalized online recommendations. The study emphasized that many of the exercise routines were performed randomly or intensely without proper technique, increasing the likelihood of injury. Furthermore, much of the online content was produced by unqualified influencers, further compounding the risks.

In this context, the development of automated systems capable of detailed, real-time motion analysis using non-invasive and cost-effective technologies becomes increasingly relevant. While promising, the implementation of machine learning (ML) models for exercise validation still faces challenges in accuracy, data quality, and integration into existing practices. Establishing standardized protocols and frameworks is crucial for ensuring reliable, interdisciplinary applications [Roggio *et al*., 2024].

Traditionally, human movement analysis relied on high-cost systems involving high-speed cameras, reflective markers, and motion sensors. Intermediate solutions like Microsoft Kinect [Newcombe *et al*., 2011] represented a leap forward by enabling markerless motion capture through RGB and infrared sensors. However, such systems still posed limitations in terms of portability, cost, and ease of deployment.

Recent advancements in computer vision and deep neural networks have enabled real-time pose estimation using only conventional cameras. Models like BlazePose [Bazarevsky *et al*., 2020] achieve high accuracy in detecting body keypoints even on mobile devices, eliminating the need for specialized hardware and allowing for more accessible and non-intrusive analysis.

Given this scenario, this work presents a web-based solution for real-time automated exercise assessment powered by machine learning. Focusing on the high plank exercise as a case study, the approach covers data extraction, transformation and loading (ETL), the training of exercise validation models and their integration into a prototype web application for real-time evaluation. By bridging technical innovation with practical applicability, this solution aims to contribute to the development of smarter and more inclusive tools for remote physical activity monitoring.

The remainder of this paper is organized as follows: Section 2 discusses related work involving pose estimation and ML-based exercise assessment. Section 3 outlines the theoretical foundations that support this research. Section 4 describes the DSR paradigm and how it guided the development and evaluation of the proposed solution. Section 5 presents the main results obtained, covering model evaluation, application performance, and proposed software design. Finally, Section 6 provides concluding remarks and suggests directions for future work.

## 1.1 Ethical issues

This research utilized human images for AI model training from two primary sources: a publicly available repository on the Kaggle platform and images of the authors themselves, obtained with appropriate consent. The images and code are available in a public GitHub repository to ensure research reproducibility.

The Kaggle dataset `Yoga For All` [Fong and Otto, 2025] was distributed under the MIT License, which permits unrestricted use, distribution, and modification, provided that the original copyright notice and license terms are included.

No personal or sensitive data were used in this research. All images employed are either publicly available or obtained with explicit consent from the authors, and none contain identifiable information or metadata that could be linked to individual identities.

A total of 363 images were used in this study. Of these, 333 images were sourced from the Kaggle dataset while the remaining 30 images were captured from the authors.

The dataset comprises 285 images of individuals identified as female and 78 images identified as male. This gender imbalance is primarily attributable to the characteristics of the Kaggle dataset, which focuses on yoga practice and predominantly features female participants, reflecting common trends in publicly available yoga-related image datasets. The inclusion of images from the authors, which exclusively depict male subjects, contributed to a modest increase in gender diversity within the dataset. Nevertheless, this imbalance remains a limitation of the present study, as the predominance of female samples may compromise the generalizability of the trained models to male users.

The Kaggle dataset used in this study does not provide detailed demographic metadata, such as age, body type, or other anthropometric attributes. The absence of this information limits the possibility of a more comprehensive demographic characterization of the dataset. Inferring or annotating such attributes would require specific resources, professional expertise, and ethical approval, which were beyond the scope of this research. Since the primary focus of this study is not the training of novel pose estimation models, but rather the end-to-end integration and deployment of existing machine learning models within a web-based exercise assessment system, a deeper demographic composition analysis was not conducted.

All pose estimation and classification processes are executed locally in the user's web browser. The trained models run entirely on the client side, and no images, video frames, or extracted pose data are transmitted to external servers or stored persistently. This enhances security and privacy by ensuring that user data remains confined to the local execution environment, minimizing the risks associated with data leakage, unauthorized access, or misuse of personal visual information.

This study presents a proof-of-concept prototype intended to serve as a foundation for future web-based applications in automated exercise assessment. It must not, under any circumstance, replace the assessment or supervision of certified health and sports professionals.

# 2 Related Work

Several studies have explored the use of computer vision techniques for exercise detection and/or assessment based on pose estimation. Some approaches validate exercises by checking whether joint angles fall within predefined thresholds, while others rely on machine learning (ML) techniques to assess more complex movements.

Kwon and Kim [2022] proposed a program designed to guide users in performing squats and push-ups with the correct posture. The system estimates the user's body landmarks from real-time webcam footage and calculates joint angles to determine whether the exercise is being executed correctly. If the angles fall outside the predefined ranges, the system provides feedback to help the user adjust their posture accordingly.

G *et al*. [2023] developed a workout trainer using pose estimation models that process the user's live video feed to extract body landmarks. The system calculates the angles formed by body parts during push-up and bicep curl exercises and compares them against expected ranges for each movement. Feedback is then provided based on this comparison, guiding the user on how to improve their form.

Sugawara [2022] introduced a pose estimation-based solution that extracts human body landmarks and uses them as input to a neural network model to classify exercise types: squats, pull-ups or push-ups. Exercise assessment is performed by calculating the error between ideal angles and those extracted from the user's movement. Once the error exceeds a certain threshold, the system provides corrective feedback to improve posture.

Ko *et al*. [2024] conducted a comparative analysis of various ML and deep learning algorithms for posture classification in bench press, squat, and deadlift exercises. Their approach stands out for providing personalized feedback in real time that helps users adjust their form. The ML models used include logistic regression, ridge classification, random forest, and gradient boosting. In addition, several deep learning solutions were developed, such as fully connected neural networks (FCNN), MLP-Mixer, and Transformer architectures. This study combines angle-based assessment with ML-driven analysis, as well as text-to-speech (TTS) technology, to enrich the feedback provided to users.

Pires *et al*. [2025] proposed an automated ergonomic assessment system that leverages human pose estimation for real-time posture analysis in industrial environments. Their approach combines YOLO[1] models for person detection with BlazePose for extracting 2D and 3D body landmarks from RGB video streams captured by surveillance cameras. Based on the estimated joint positions, the system computes Rapid Upper Limb Assessment (RULA) scores to quantify ergonomic risks associated with workers' postures. The authors demonstrate that computer vision–based pose estimation can provide an objective and scalable alternative to traditional observational ergonomic assessments, which are often subjective and labor-intensive.

In addition to individual system proposals, systematic

[1] You Only Look Once (YOLO) is a real-time object detection system that uses a single deep learning model to find and classify multiple objects in an image or video by looking at the whole image just once.

analyses of the literature have sought to consolidate knowledge on the application of pose estimation for exercise assessment. Difini *et al*. [2021] conducted a systematic literature review focusing on pose estimation techniques applied to sports and exercise training scenarios, analyzing studies published between 2011 and 2021. Their review highlights a progressive shift from traditional feature-based methods to deep learning–based approaches, which significantly improved keypoint detection accuracy in 2D images and videos. However, the authors also identify persistent challenges, including motion blur, self-occlusion, fast and complex movements, and the limited availability of domain-specific datasets.

Similarly, Dias *et al*. [2023] also carried out a literature review focusing on the application of human pose estimation techniques to physical exercise analysis, with particular emphasis on calisthenics training. Motivated by the increased practice of at-home and outdoor exercises during the COVID-19 pandemic, the authors reviewed studies that employed pose estimation libraries such as OpenPose and MediaPipe to assist users in posture analysis and movement correction. The authors concluded that human pose estimation represents a powerful tool for exercise guidance, highlighting its effective application in health-related contexts such as physical rehabilitation, education and interactive systems.

Despite the increasing sophistication of automated exercise validation systems, recent studies have highlighted important limitations in current pose estimation models when applied to health or clinical contexts. Moreira *et al*. [2024], for instance, evaluated the performance of several monocular pose estimation models integrated into a mobile application for assessing upper limb range of motion (ROM). Their results demonstrated that although models such as MoveNet Thunder INT16 showed promising accuracy in certain shoulder movements, errors greater than 10° were common, particularly in elbow flexion. The authors emphasize that, despite the potential of pose estimation models to facilitate quantitative assessments, their current accuracy remains insufficient for reliable clinical use without professional oversight. These findings underscore the importance of combining automated analysis with expert supervision.

Gonçalves *et al*. [2024] conducted a comprehensive review of computer vision techniques applied to the detection and correction of physical exercise postures. The authors highlight that, despite the technological advances in body pose detection and movement tracking, most existing approaches remain fragmented, focusing on specific algorithmic aspects rather than integrated and accessible solutions. They identify several persistent challenges, such as the high computational demands of pose estimation models, difficulties in handling occlusions, and inconsistencies caused by different camera hardware and lighting conditions. The authors concluded that there is a clear concern with algorithms and their accuracy, but an application through an integrated app that allows users with a mobile device to perform their exercises safely anywhere with specific postures is still less visible.

The proposed application in this work aims to automate physical exercise validation systems, drawing inspiration from the approaches found in the literature but also detailing how these models can be effectively employed in broadly available environments, such as modern web browsers. Rather than focusing solely on proposing new classification models, this work emphasizes the end-to-end integration of existing machine learning techniques into a modular, scalable, and browser-compatible solution. This practical perspective addresses the often-overlooked challenges of deployment, performance, and usability, thus bridging the gap between academic research and real-world application in digital health technologies.

# 3 Theoretical Framework

This section presents the fundamental concepts and technologies employed in this work.

## 3.1 Pose Estimation

Pose estimation is a computer vision technique that involves identifying and locating human body joints or parts from images or videos. The primary goal is to accurately estimate the position of landmarks – such as wrists, elbows, knees, and ankles – that together form a structural representation of the human skeleton in 2D or 3D points [Zheng *et al*., 2023].

This technique has gained wide applicability in fields such as digital animation, human-computer interaction, motor rehabilitation, sports analysis, and physical activity monitoring. One of the major advantages of modern pose estimation is its ability to run in real time using only standard RGB cameras, without the need for depth sensors or wearable devices.

MediaPipe is an open-source framework developed by Google that provides a collection of machine learning pipelines for real-time perception tasks. Its pose estimation pipeline is a variant of the BlazePose model [Bazarevsky *et al*., 2020], which incorporates the Generative Human Model (GHUM) [Xu *et al*., 2020], a 3D human shape modeling pipeline, to estimate a person's full body 3D pose from images and videos.

This solution, which we will refer to simply as BlazePose, is capable of computing the 3D coordinates of 33 human body landmarks and returns two types of coordinates: normalized landmarks and world landmarks.

Normalized landmarks refer to coordinates relative to the image dimensions. The $x$ and $y$ coordinates range from $[0, 1]$, representing the relative position with respect to the top left corner of the image (e.g., points (1, 1) and (0.5, 0.5) correspond to the bottom right corner and the center of the image, respectively). The $z$ coordinate represents relative depth (distance from the camera), but not in real-world scale. These points are especially useful for 2D visualization overlaid on the original image.

World landmarks, on the other hand, refer to absolute coordinates in a 3D space, expressed in meters, with the origin of the reference system positioned at the center of the hips. This system is more suitable for kinematic analysis and more accurate spatial measurements.

## 3.2 Angle Calculation

The use of joint angles as a human-meaningful representation of kinematic data is particularly promising for applications where interpretability and dialogue with human experts are important, such as in many sports and medical applications [Schlegel *et al*., 2024]. Given three 3D points obtained, for example, by using pose estimation models such as BlazePose,

it is possible to compute the angle between the vectors connecting these points.

Using joint angles instead of raw point coordinates offers several advantages, such as reduced dimensionality, invariance to translation and scale, and improved generalizability across different body types and initial positions. This compact and robust representation has been successfully employed in several studies involving human motion analysis.

Given three points $\mathbf{p}_1$, $\mathbf{p}_2$, and $\mathbf{p}_3$, the following vectors can be defined:

$$\mathbf{u} = \mathbf{p}_2 - \mathbf{p}_1$$

$$\mathbf{v} = \mathbf{p}_3 - \mathbf{p}_2$$

The angle $\theta$ between these vectors, in radians, is defined as:

$$\theta = \arccos\left(\frac{\mathbf{u} \cdot \mathbf{v}}{\|\mathbf{u}\|\|\mathbf{v}\|}\right), \quad \theta \in [0, \pi] \tag{1}$$

## 3.3 Classification Models

Classification models (or simply classifiers) are a type of machine learning model used to categorize data (commonly referred to as tuples[2]) into predefined classes. The training process of a classifier involves learning a mapping function $f : X \to y$ from a labeled dataset, where each tuple is associated with a known class label (the terms *class* and *label* are often used interchangeably to denote the categorical outcome that the model aims to predict). Once trained, the classifier can predict labels for unseen data. These models are widely applied in domains such as fraud detection, targeted marketing, performance prediction, manufacturing, and medical diagnosis [Han *et al.*, 2011].

One commonly used technique to improve the accuracy of classification models is the use of ensemble methods. An ensemble model is a composite model composed of multiple individual classifiers, whose predictions are combined to produce a more accurate and robust final decision. Ensemble methods are generally more accurate than their individual component classifiers, as they leverage the strengths and diversity of each model to reduce bias and variance [Han *et al.*, 2011].

## 3.4 Evaluation Metrics

The evaluation of supervised machine learning models, particularly for classification tasks, requires statistical metrics that quantify prediction quality. Commonly used metrics are accuracy, precision, recall, and F1-score.

These metrics are derived from the confusion matrix, a table that summarizes the number of correct and incorrect classifications, allowing for a detailed analysis of how the model performs across different classes. Table 1 shows the structure of a confusion matrix for binary classification.

The predictions are organized into four categories:

- True Positive ($TP$): number of positive samples correctly classified as positive;
- True Negative ($TN$): number of negative samples correctly classified as negative;
- False Positive ($FP$): number of negative samples incorrectly classified as positive (false alarms);

[2]A tuple is an ordered collection of elements that can contain different types of data.

| | Actually Positive (1) | Actually Negative (0) |
|---|---|---|
| Predicted Positive (1) | True Positives (TPs) | False Positives (FPs) |
| Predicted Negative (0) | False Negatives (FNs) | True Negatives (TNs) |

**Table 1.** Binary confusion matrix (adapted from Draelos [2019])

- False Negative ($FN$): number of positive samples incorrectly classified as negative (missed detections).

Based on these categories, the metrics are defined as follows:

- Accuracy: Represents the proportion of correct predictions—both positive and negative—relative to the total number of evaluated samples. It is a general performance metric and is especially useful when classes are balanced.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{2}$$

- Precision: Measures the reliability of positive predictions, i.e., the proportion of instances predicted as positive that are truly positive. This metric is particularly important in scenarios where false positives are undesirable.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{3}$$

- Recall: Also known as sensitivity or true positive rate, this metric quantifies the model's ability to correctly identify all positive instances.

$$\text{Recall} = \frac{TP}{TP + FN} \tag{4}$$

- F1-score: Represents the harmonic mean between precision and recall. It is especially useful when classes are imbalanced or when both false positives and false negatives are relevant.

$$\text{F1-score} = 2 \cdot \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{5}$$

## 3.5 ONNX

The Open Neural Network Exchange (ONNX[3]) is an open standard format developed by Microsoft and Meta to enable seamless interoperability between various machine learning frameworks. Its primary goal is to allow models trained in one framework (such as PyTorch, TensorFlow, Keras, or scikit-learn) to be exported and used across different platforms and runtimes without the need for retraining or complex conversions.

ONNX models are represented using a computational graph structure, where each node corresponds to a specific operation (e.g., matrix multiplication, convolution, activation functions) and each edge represents the data flow between

[3]`https://onnx.ai/`

operations. This graph-based representation makes the format highly portable and optimizable.

A key component of the ONNX ecosystem is the ONNX Runtime, a high-performance inference engine optimized for cross-platform deployment. It supports hardware acceleration through providers such as CUDA (for NVIDIA GPUs), DirectML, TensorRT, and OpenVINO, making it suitable for use on both cloud infrastructure and edge devices.

Another relevant aspect of ONNX is the availability of libraries for web-based inference, such as ONNX.js and ONNX Runtime Web. These JavaScript-based tools allow pre-trained ONNX models to be executed directly in the browser, without requiring server-side computation. This client-side capability reduces latency, enhances scalability, and improves data privacy, since user data does not need to leave the local device. Additionally, web-based inference facilitates broader accessibility, as it only requires a modern browser and does not depend on specific operating systems or hardware configurations.

# 4 Materials and Methods

This study constitutes an applied and exploratory research effort, aiming to investigate and demonstrate the training and integration of machine learning models within a web application capable of performing real-time automated assessment of physical exercise execution. In addition to its practical application, the study also proposes a modular and scalable software design that facilitates such integration. Thus, it contributes both to technological innovation and methodological advancement in the development of accessible tools for remote monitoring of physical activities.

The Design Science Research (DSR) paradigm was adopted as the central methodology for the development and evaluation of the technological artifact, following a systematic approach to solving practical problems in the domain of automated exercise assessment [Peffers *et al*., 2007].

The research process was structured into six stages, adapted to the context of this study: (1) problem identification; (2) definition of solution objectives; (3) design and development; (4) demonstration; (5) evaluation; (6) communication of results.

In Steps 1 and 2, the problem was identified and the solution objectives were defined based on an analysis of the research context and the identification of gaps in both the scientific literature and existing technological solutions. This process involved a literature review using databases such as Google Scholar and ResearchGate, supplemented by an exploration of open-source repositories, commercial fitness applications, and previous studies on pose estimation and exercise assessment. The review showed a lack of accessible, browser-based tools capable of providing automated and immediate feedback on exercise execution using pose estimation techniques. In light of this context, the objective was to develop a lightweight, modular, and real-time web application capable of validating physical exercises through pose-based inference, integrating multiple validation strategies while ensuring ease of use and extensibility.

Stage 3 involved the development of the artifact across two main fronts. First, the training of the machine learning models was conducted using a dataset composed of 363 images of the target exercise, collected from two primary sources: the publicly available Kaggle repository Yoga For All [Fong and Otto, 2025] and images produced by the authors themselves. The Kaggle dataset provided pre-labeled samples organized into the folders `correct` and `incorrect`. The images captured by the authors were labeled by the authors themselves, with incorrect samples obtained through deliberately misperformed executions designed to reproduce common posture errors, and correct samples following established biomechanical guidelines for proper execution. The models were trained using 16 angles computed from body landmarks extracted via BlazePose as input features. The evaluated classifiers included Gradient Boosting, Logistic Regression, Random Forest, Support Vector Machine (SVM), and a Fully Connected Neural Network (FCNN). All trained models were exported to the `.onnx` format for integration into the web application, and their performance was evaluated using the metrics described in Section 3.4.

The second front focused on the development of a modular and extensible web application implemented using the React framework. The application features a responsive user interface that allows users to select exercises and models, initiate camera capture, and receive real-time visual feedback. It integrates BlazePose for in-browser pose estimation and performs local classification using the selected machine learning model. The software architecture is centered on shared interfaces that abstract validator behavior, enabling the seamless integration of multiple validation strategies, such as machine learning models, empirical rules, or ensemble methods. This design promotes separation of concerns and adheres to software engineering best practices, supporting maintainability, scalability, and reusability.

In Stage 4, the solution was demonstrated through controlled tests conducted by the authors themselves, who performed the high plank exercise in front of the camera to verify the application's ability to accurately capture body movements and provide the expected real-time feedback. This step allowed for validation of the full application pipeline, from pose estimation to exercise classification.

Stage 5 consisted of evaluating the application's functionality based on a qualitative analysis carried out by the authors. Different variations of the exercise were intentionally performed to simulate both correct and incorrect postures. In addition, inference times were measured for each classification model across different devices, including both desktop and mobile environments, in order to assess the performance and responsiveness of the application under varying hardware conditions. The evaluation focused on the system's ability to distinguish these variations accurately and provide consistent and timely feedback in accordance with the model outputs.

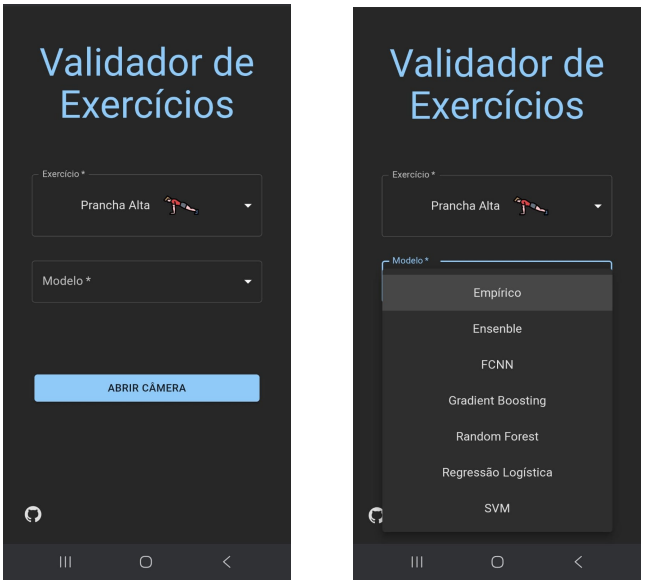
Finally, Stage 6 is represented by this publication, which aims to communicate the proposal, development process, challenges encountered, and the results obtained, contributing to the growing body of literature on the application of machine learning and computer vision techniques to the assessment of physical exercises.

# 5 Results

This section presents the resulting web application with the integrated models, in addition to the proposed software design. It also details the entire model training pipeline: from data collection, labeling, and preprocessing to the definition of model architectures and hyperparameters and training results.

## 5.1 Web Application

The web application was developed using the React framework, in combination with the `@mediapipe/tasks-vision` package, which incorporates the BlazePose model for pose estimation. In the main interface, shown in Figure 1a, the user can select the exercise to be evaluated. Once selected, the "Modelo" ("Model" in Portuguese) dropdown menu becomes available, listing the classification models for that exercise, as illustrated in Figure 1b.




**(a)** Web app main page **(b)** Model selection

**Figure 1.** Screenshots of the main page

By clicking the "ABRIR CÂMERA" button ("OPEN CAMERA" in Portuguese), the application initiates the loading process of the camera, the BlazePose model and the selected classification model. Once all components are loaded, the interface begins displaying the camera feed in real time along with the inference result based on the extracted body landmarks.

If no body is detected by BlazePose, the application displays the message "aguardando posição" ("waiting pose" in Portuguese), as seen in Figure 2a. As soon as a body is detected, its landmarks are processed, and the classification result is shown to the user, as shown in Figures 2b, 2c and 2d.

Note that, in the case of negative feedback from the empirical model, unlike the other classifiers, it is possible to provide more specific guidance about what is incorrect in the execution because this model evaluates each joint angle individually against predefined threshold values, allowing it to identify which specific movement parameters deviate from the expected range. This is seen in Figure 2d, where the message "Mantenha os punhos alinhados os ombros e quadris" ("Keep the wrists aligned with the shoulders and hips " in Portuguese) is displayed instead of simply "incorrect".

Unlike the empirical model, the ML models present a limitation regarding the specificity of the feedback provided. Since these models were trained using only binary labels (`correct` and `incorrect`), they are unable to identify which particular aspect of the movement caused an incorrect classification. This restricts the precision of the feedback delivered to the user. Such limitation could be mitigated by training models with more detailed labels or by adopting hybrid approaches that combine machine learning models with rule-based models, such as the empirical model, to generate richer and more informative feedback.

Furthermore, when the application is used without external assistance, the user may face difficulties in observing the feedback on the screen while performing the exercise. A potential solution, adopted by other authors such as Ko *et al.* [2024], is the inclusion of auditory feedback, allowing users to receive verbal cues in real time without interrupting the exercise execution.

## 5.2 Software Design

Figure 3 presents a high-level UML class diagram of the proposed software design. It follows the SOLID principles to ensure modularity, extensibility, and ease of maintenance.

The system is structured around a common `Classifier` interface, which abstracts the validation logic independently of the underlying implementation. This design allows different validation strategies—such as machine learning models or rule-based approaches—to be used interchangeably and combined when needed. Model management and instantiation are centralized through a factory mechanism, ensuring controlled loading and efficient resource usage.

Pose estimation and exercise validation are encapsulated in separate components, decoupling motion capture from classification logic. This separation facilitates scalability and simplifies the integration of new exercises or models.

## 5.3 Training Process

This topic describes the process of collecting, preprocessing and augmenting the training data, as well as the definition and evaluation of the trained models.

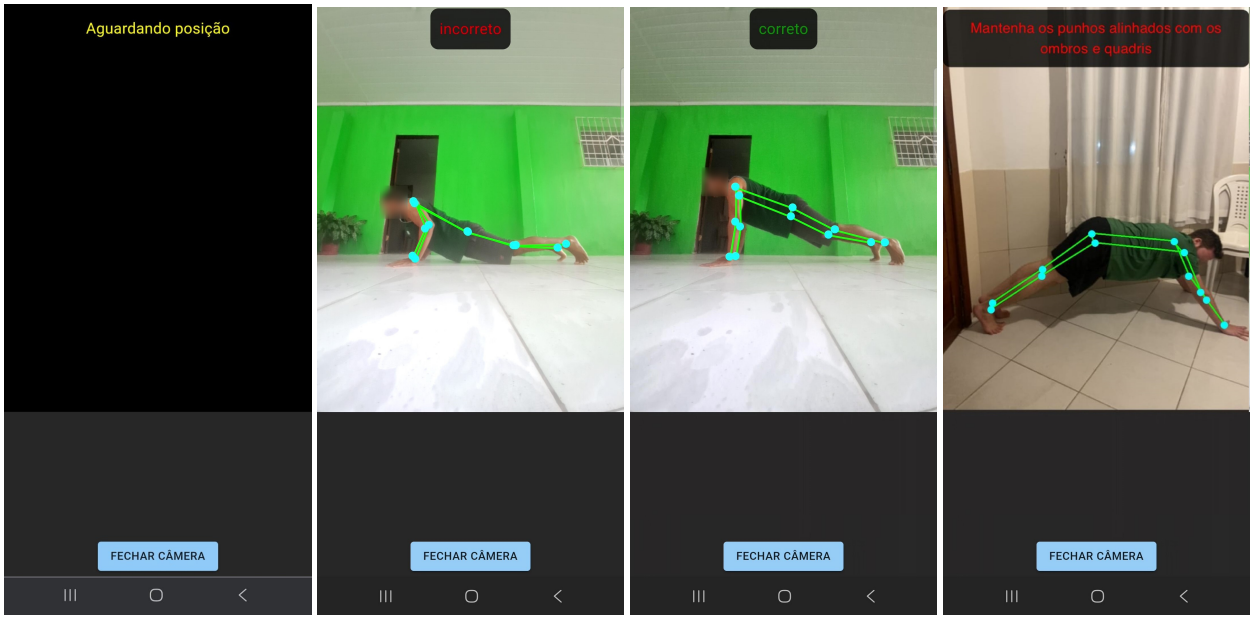
### 5.3.1 *Data Collection and Preprocessing*

A dataset was constructed by collecting 363 images representing both correct and incorrect executions of the high plank exercise. The sources used for this purpose were the Kaggle repository `Yoga For All` [Fong and Otto, 2025] and images from the authors themselves.

To ensure diversity in the dataset, images with varying resolutions, body proportions, and camera angles were selected. After curation, all images were manually labeled with the classes *correct* (correct execution) or *incorrect* (incorrect execution).

The input to the models consisted of joint angles computed from the world landmarks extracted using BlazePose. These angles were calculated using Equation 1, and subsequently normalized by dividing them by $\pi$, resulting in values $\theta_{\text{norm}} \in [0, 1]$.

Data augmentation was employed to enhance the robustness and generalizability capability of the models. For each image in the dataset, multiple variations were generated




**(a)** No pose feedback **(b)** Incorrect feedback **(c)** Correct feedback **(d)** Empirical model incorrect feedback

**Figure 2.** Exercise execution feedback

through controlled geometric transformations applied prior to landmark extraction.

Specifically, for every original image, a combination of horizontal mirroring, rotation, and scaling operations was applied according to predefined parameters. The mirroring operation was used to simulate left-right symmetry and introduce pose variability across both body sides. Rotational transformations were applied with angles of $-15°$, $-5°$, $5°$, and $15°$, while scaling factors of $0.9$ and $1.1$ were employed to mimic variations in camera distance and subject proportions.

These transformations were performed before passing each image to the BlazePose model, ensuring that the resulting sets of world landmarks reflected genuine geometric variations introduced by the augmentation. For every combination of mirroring, rotation, and scaling, landmarks were extracted and converted into a vector of joint angles.

Importantly, due to BlazePose's non-deterministic behavior under image transformations, the same anatomical points exhibit slightly different coordinates across augmented versions of the same image. This results in meaningful variability in the computed joint angles, effectively increasing the diversity of the training data.

### 5.3.2 Model Definition

The classification models used in this study were organized into two main groups:

- Non-neural models:
  - Gradient Boosting
  - Logistic Regression
  - Random Forest
  - Support Vector Machine (SVM)
- Neural network-based models:
  - Fully Connected Neural Network (FCNN)

Additionally, a non-ML-based model was incorporated into the web application to validate exercises by verifying whether the joint angles between specific body landmarks fall within predefined thresholds. These thresholds were empirically determined based on measurements extracted from images of correctly performed exercises. Due to that, this model will be referred to as the empirical model.

Another model incorporated into the web application was the ensemble model, which combined the outputs of the previously mentioned ML and empirical models. The ensemble performed a majority voting process to determine the final classification. This strategy aimed to enhance decision robustness by incorporating multiple assessment perspectives.

It is important to emphasize that the empirical and ensemble models were not developed to compete with the ML models in terms of predictive performance. Rather, their purpose was to showcase the flexibility of the proposed software design, specifically, its ability to seamlessly integrate rule-based validation strategies (commonly found in the literature reviewed in Section 2) within the same interface adopted for ML classifiers.

The models were implemented using the Python programming language. The FCNN was built and trained using the Keras[4] package. The non-neural models were implemented with the scikit-learn[5] package.

For model evaluation, the dataset was split into training and testing subsets using a 70/30 ratio through the `train_test_split` function from scikit-learn. A fixed random seed was used to ensure that all models were trained and evaluated on identical splits, allowing for a fair comparison of results.

To validate the high plank exercise, we considered joint landmarks from both sides of the body, specifically the wrists, elbows, shoulders, hips, knees, and ankles.

As discussed in Section 5.3.2, the models were trained using joint angles as input features. The 16 angles employed were calculated from the following groups of three landmarks:

[4] https://keras.io/
[5] https://scikit-learn.org/stable/

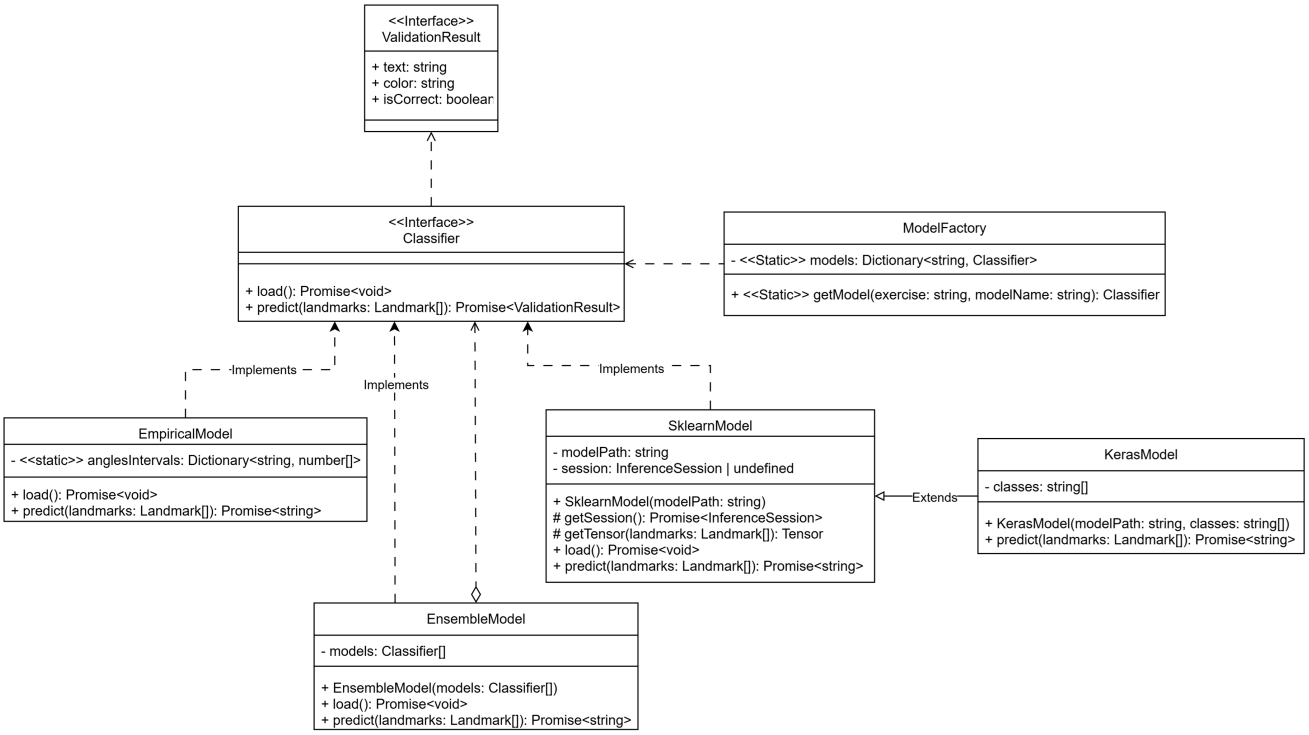



**Figure 3.** UML class diagram

1. left wrist, left elbow, left shoulder
2. right wrist, right elbow, right shoulder
3. left wrist, left shoulder, right shoulder
4. right wrist, right shoulder, left shoulder
5. left wrist, left shoulder, left hip
6. right wrist, right shoulder, right hip
7. left shoulder, left hip, left knee
8. right shoulder, right hip, right knee
9. left hip, left knee, left ankle
10. right hip, right knee, right ankle
11. Left ankle, left hip, right hip
12. right ankle, right hip, left hip
13. left foot index, left wrist, left shoulder
14. right foot index, right wrist, right shoulder
15. left foot index, left wrist, right wrist
16. right foot index, right wrist, left wrist

### *5.3.3 Hyperparameters and Model Architecture*

The selection of hyperparameters for the non-neural models was performed using a grid search with cross-validation, implemented through the `GridSearchCV` function. The specific hyperparameters used for each model are presented in Table 2.

Figure 4 illustrates the architecture of the FCNN used in this study. The input layer consists of 16 normalized angles, as described in Section 5.3.2.

Following the input layer, a dense layer with 64 neurons and ReLU activation function maps the angles into a higher-dimensional representation space. This is followed by a batch normalization layer, which normalizes the outputs of the previous layer to promote training stability and accelerate convergence.

Next, a dropout layer with a 30% dropout rate is applied to reduce the risk of overfitting by randomly deactivating 30% of the neurons during each training epoch. The architecture continues with a second dense layer containing 32 neurons, also followed by batch normalization and dropout layers (30%).

The final layer is a dense layer with two units and a sigmoid activation function, outputting the probabilities associated with the *correct* and *incorrect* classes.

**Table 2.** Hyperparameters used in grid search for non-neural models

| | |
|---|---|
| **Gradient Boosting** | |
| learning_rate | 0.01, 0.1, 0.5, 1 |
| n_estimators | 50, 100, 200, 300, 500 |
| subsample | 0.6, 0.8, 1.0 |
| min_sample_split | 2, 5, 10, 20, 50 |
| max_depth | 3, 5, 10 |
| **Logistic Regression** | |
| C | 0.01, 0.1, 1, 10, 50, 100 |
| penalty | None, l1, l2, elasticnet |
| solver | lbfgs, liblinear, newton-cg, newton-cholesky, sag, saga |
| max_iter | 10, 100, 1000, 10000 |
| **Random Forest** | |
| n_estimators | 10, 50, 100, 200 |
| criterion | gini, entropy, log_loss |
| max_depth | None, 10, 20, 50 |
| min_samples_split | 2, 5, 10 |
| min_samples_leaf | 1, 2, 5, 10 |
| **SVM** | |
| C | 0.01, 0.1, 1, 10, 50 |
| kernel | linear, poly, rbf, sigmoid |
| degree | 2, 3, 4, 5, 10 |
| gamma | scale, auto |

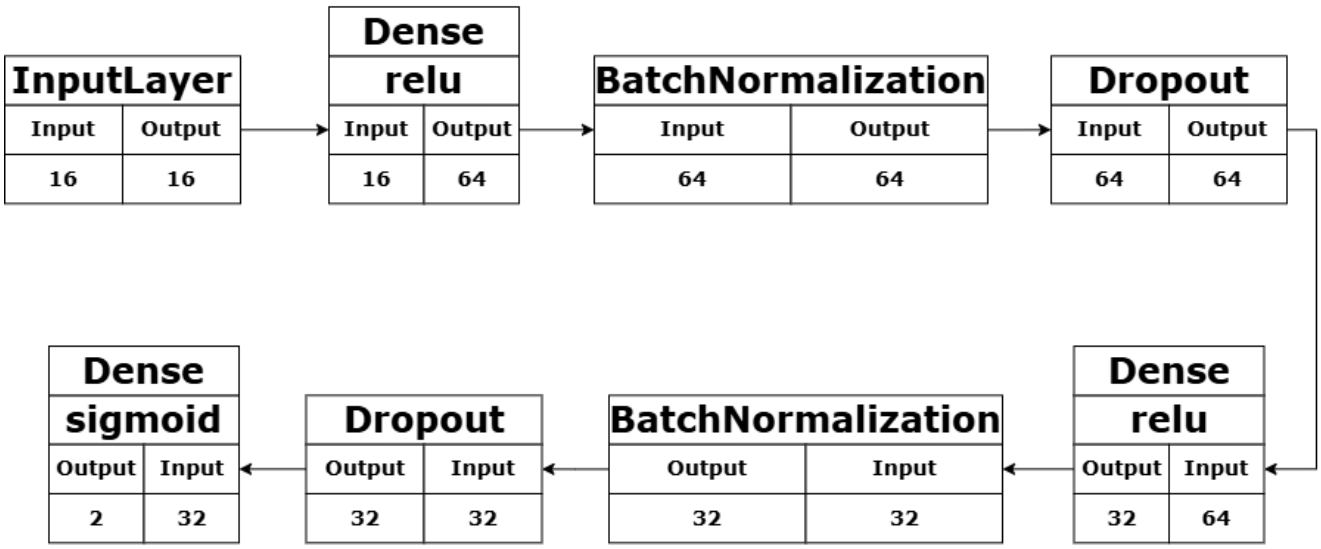



**Figure 4.** FCNN model diagram

The model was compiled using the Adam optimizer with an initial learning rate of $10^{-3}$. The loss function adopted was binary cross-entropy, which is appropriate for binary classification tasks with probabilistic outputs.

To improve training efficiency and prevent overfitting or convergence stagnation, the following callbacks were employed:

- EarlyStopping: Automatically halts training if the validation loss does not improve for 50 consecutive epochs. With the parameter `restore_best_weights=True`, the model restores the weights corresponding to the epoch with the lowest validation loss.
- ReduceLROnPlateau: Automatically reduces the learning rate in response to validation loss stagnation, triggered after 30 epochs without improvement. This helps fine-tune convergence in the later stages of training.

#### 5.3.4 *Training Results*

The models were trained using 3768 samples labeled as *correct* and 3769 samples labeled as *incorrect*. They were evaluated using a test set of 108 samples, consisting of 38 labeled as *correct* and 70 as *incorrect*. The random seed used in the `train_test_split` function was 993139. Overall, the models performed similarly, achieving accuracies above 95%. This result was expected, as the plank exercise is easier to validate due to the body remaining static.

The best hyperparameters found for the non-neural models among those defined in Section 5.3.3 are presented in Table 3.

For the FCNN model, we set a batch size of 32 and a maximum of 1000 epochs for training. Due to the callbacks described in Section 5.3.3, the training was stopped early after 80 epochs. The accuracy and loss curves are shown in Figure 5.

The graphs indicate rapid convergence within the first few epochs and no significant signs of overfitting or underfitting, as shown by the convergence and proximity between training and validation metrics. A comparison of model performance is presented in Table 4.

**Table 3.** Best hyperparameters found for each model

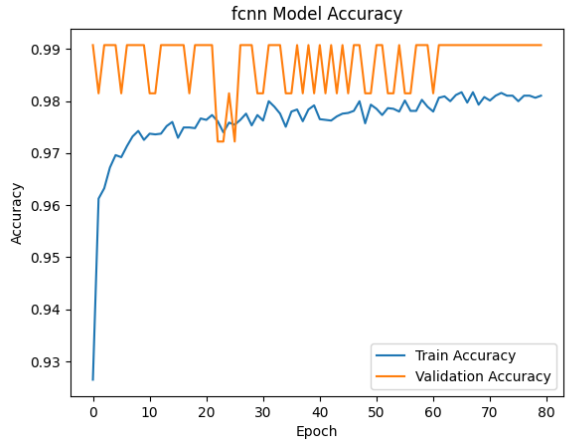
| | |
|---|---|
| **Gradient Boosting** | |
| learning_rate | 0.1 |
| n_estimators | 500 |
| subsample | 0.8 |
| min_sample_split | 10 |
| max_depth | 5 |
| **Logistic Regression** | |
| C | 10 |
| penalty | l1 |
| solver | saga |
| max_iter | 10 |
| **Random Forest** | |
| n_estimators | 10 |
| criterion | entropy |
| max_depth | None |
| min_samples_split | 5 |
| min_samples_leaf | 2 |
| **SVM** | |
| C | 1 |
| kernel | rbf |
| degree | 2 |
| gamma | scale |

**Table 4.** Comparison of training results with test set

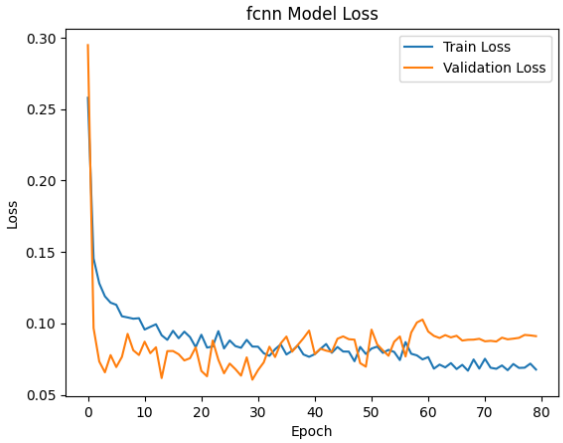
| Model | Accuracy | Precision | Recall | F1-score |
|---|---|---|---|---|
| FCNN | 0.9815 | 0.9797 | 0.9797 | 0.9797 |
| Grad. Boost. | 0.9907 | 0.9872 | 0.9929 | 0.9899 |
| Log. Regr. | 0.9815 | 0.9797 | 0.9797 | 0.9797 |
| Rand. For. | 0.9907 | 0.9872 | 0.9929 | 0.9899 |
| SVM | 0.9815 | 0.9797 | 0.9797 | 0.9797 |

## 5.4 Web Application Evaluation

To assess the models' behavior, the authors conducted tests by performing the target exercise both correctly and incorrectly, as well as varying specific joint angles to simulate borderline cases. These tests allowed the evaluation of how each model responded to different execution patterns, helping to




**(a)** FCNN accuracy chart

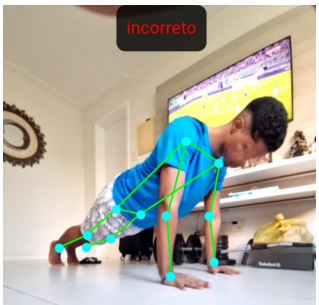



**(b)** FCNN loss chart

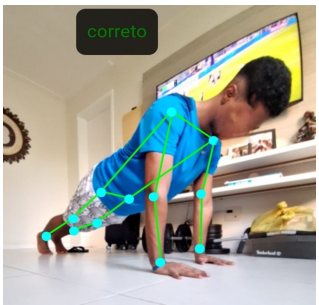
**Figure 5.** Training performance of the FCNN model

identify strengths and limitations in their classification strategies. It is important to note that the evaluation did not involve professionals from the fields of physical education, physiotherapy, or sports science, which would have been ideal to ensure biomechanical correctness and safety. However, this limitation was considered acceptable within the scope of the present work, as the primary goal was not to develop and validate the exercise evaluation itself, but rather to demonstrate the complete process of training, deploying, and integrating machine learning models into a web-based application.

During testing, it was observed that, at certain angles and distances from the camera, correct executions were occasionally classified as `incorrect`. Interestingly, when the camera position was slightly adjusted, the same execution was often reclassified as correct by the model. This behavior, illustrated in Figure 6, suggests that small variations in certain viewpoints can influence landmark detection and, consequently, the classification outcome.




**(a)** Correct execution misclassified as incorrect




**(b)** Same execution correctly classified after slight camera adjustment

**Figure 6.** Example of a false negative classification influenced by camera viewpoint

To further validate the web application, the Gradient Boosting model was tested using a dataset of 40 images, 20 correct and 20 incorrect, collected from the authors themselves at different camera positions in the web application. This model was selected among the others because it achieved the highest accuracy during the training stage. The objective of this evaluation was to verify whether the performance observed in the training would translate into consistent results within the real-time web-based environment.

The obtained results are summarized in the confusion matrix presented in Table 5. Overall, the model achieved an accuracy of 85%, correctly identifying the majority of both correct and incorrect executions. However, a relatively larger number of false incorrect classifications was observed, where the model marked an execution as `incorrect` despite it being performed correctly. Those results are consistent with the observations that certain camera angles and distances occasionally led the model to classify a correct execution as `incorrect`.

**Table 5.** Confusion matrix of Gradient Boosting model in web application testing

| | Predicted Correct | Predicted Incorrect |
|---|---|---|
| Actual Correct | 15 | 5 |
| Actual Incorrect | 1 | 19 |

This limitation can be mitigated through two complementary strategies. First, by enriching the training dataset with images captured from a wider variety of favorable viewpoints, the models can learn to become more robust to variations in camera positioning. Second, the application itself can provide guidance to the user, suggesting recommended camera angles and distances that maximize the accuracy of landmark detection.

Another analysis carried out was the inference time of the models, which was measured across the following devices: Samsung Galaxy A3 Core (I), Samsung Galaxy A53 5G (II), Samsung Galaxy A54 5G (III), Apple iPhone 11 (IV). Table 6 summarizes the average inference times and standard deviations obtained for BlazePose on each evaluated device.

**Table 6.** BlazePose inference time (mean $\pm$ std. deviation)

| Device | Time (ms) |
|---|---|
| I | $1020.8000 \pm 9.0308$ |
| II | $129.0200 \pm 13.5378$ |
| III | $94.2200 \pm 14.3955$ |
| IV | $51.1500 \pm 5.7643$ |

Table 7 presents the average inference times and stan-

dard deviations for the classification models trained in this work, also measured on the evaluated devices. For Device IV, inference times are only reported for the empirical model due to a limitation observed during testing: models that required loading a `.onnx` file remained indefinitely in the loading state on iOS. This behavior suggests a possible error in the ONNX file loading process, since the empirical model, which does not depend on external `.onnx` files, was executed without issues.

All models show significantly lower inference times than the time required to extract landmarks using BlazePose. This indicates that model inference is not a performance bottleneck in the complete real-time analysis pipeline.

Device I, being a lower-end smartphone with more limited computational resources, exhibited the lowest overall performance among the tested devices. Although the classification models showed higher inference times compared to the other devices, they were still able to execute within a timeframe compatible with real-time use. The main performance bottleneck was observed in the BlazePose inference stage, which required approximately 1 second per frame on this device. This latency implies that the system would take over one second to provide feedback to the user, significantly reducing the responsiveness of the application. These results highlight that, despite the classification stage being computationally lightweight, real-time pose-based validation may still be constrained on less capable devices due to the higher processing demands of the landmark extraction stage.

The empirical model is naturally the fastest, given the simplicity of its validation strategy. Additionally, it has a key advantage in terms of interpretability: by identifying which angles fall outside the expected ranges, it becomes possible to directly infer which joints are misaligned, facilitating the generation of specific feedback for the user.

Consequently, this type of model may be preferable for isometric exercises, such as the high plank, where the joint angles remain within well-defined ranges, making it sufficient to correctly define these thresholds. It can also be ideal for verifying whether joints are within expected angle intervals, while delegating the validation of joint movements and dynamic patterns to more robust models, such as those based on machine learning.

# 6 Conclusion

This work presented the development of a web application prototype for automated and real-time assessment of exercise execution, integrating machine learning models with pose estimation directly in the browser. The system design allows for scalable integration of new exercises and classifiers, ensuring flexibility and maintainability.

From a broader perspective, this study represents an advancement towards the universality and accessibility of physical activity monitoring. By demonstrating a complete process for developing web-based applications for automated exercise validation, it expands the applicability of such technologies beyond controlled or specialized environments. While several related works have proposed similar validation systems in desktop or standalone applications, the proposed prototype leverages the Web as an execution environment, requiring no installation and being compatible across devices and operating systems, thus contributing to more inclusive and readily deployable digital health tools.

Ensemble models were highlighted as a promising approach, enabling strategies such as model specialization by body segment, early stopping on critical errors, or weighted voting schemes. These extensions point to the potential of evolving the application into more intelligent and robust systems.

The high plank exercise was chosen as a case study due to its isometric nature and easier validation. The assessment was intentionally simplified to a binary output (`correct/incorrect`), as the focus of this work was to demonstrate the end-to-end pipeline of data collection, training, and web integration, rather than optimizing exercise validation per se.

The inability to load certain ONNX-based models on iOS devices revealed a compatibility issue that must be addressed to guarantee broader accessibility. As future work, we envision extending the application to more complex exercises and exploring related domains such as gesture recognition and sign language.

We acknowledge several limitations concerning the dataset and methodology. The experiments were conducted in a controlled setting with executions performed only by the authors, without external validation from sports professionals. Additionally, the training data was small and did not consider demographic or phenotypic diversity, which may introduce biases and reduce the model's generalizability across different populations. These limitations were tolerated given the exploratory and demonstrative scope of this project, however, they must be addressed in future research and in any user-oriented applications to ensure reliability, inclusiveness, and safety.

Finally, it is important to emphasize that this system, both in its current prototype form and in potential future commercial or research versions, is not intended to replace the supervision of qualified professionals. Instead, it should be used as a complementary tool to support physical educators and health practitioners in promoting safer and more effective exercise execution.

# Declarations

## Acknowledgements

AI tools were used to assist in the text review and translation.

## Funding

This research did not receive any direct or indirect funding.

## Authors' Contributions

**R. Santos contributions:** Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Project administration, Software, Supervision, Visualization, Writing – original draft, Writing – review & editing.

**A. Sant'anna contributions:** Conceptualization, Investigation, Methodology, Software, Validation, Writing – original draft.

**L. Machado contributions:** Conceptualization, Investigation, Methodology, Software, Validation, Writing – original draft.

**L. Santos contributions:** Conceptualization, Investigation, Methodology, Software, Validation, Writing – original draft.

**Table 7.** Models inference time (mean ± std. deviation)

| Model | I | II | III | IV |
|---|---|---|---|---|
| Empirical | $0.4400 \pm 0.1020$ | $0.1950 \pm 0.0589$ | $0.1500 \pm 0.0671$ | $0.1500 \pm 0.1879$ |
| Ensemble | $9.7611 \pm 7.6983$ | $1.9950 \pm 0.4364$ | $2.0150 \pm 0.9520$ | – |
| FCNN | $2.3214 \pm 0.9615$ | $0.9600 \pm 0.2131$ | $0.6842 \pm 0.1424$ | – |
| Gradient Boosting | $1.9533 \pm 0.1668$ | $0.9440 \pm 0.1655$ | $0.7300 \pm 0.3466$ | – |
| Logistic Regression | $1.9294 \pm 0.3232$ | $0.8250 \pm 0.1609$ | $0.6200 \pm 0.1166$ | – |
| Random Forest | $1.9300 \pm 0.6596$ | $0.9000 \pm 0.2121$ | $0.7250 \pm 0.2165$ | – |
| SVM | $2.1846 \pm 0.4588$ | $0.8850 \pm 0.1905$ | $0.8071 \pm 0.5885$ | – |

**M. Soussa contributions:** Methodology, Validation, Writing – review & editing.

## Competing interests

The authors declare that they have no competing interests.

## Availability of data and materials

The Python code and images dataset used in training are available in the following repository: `https://github.com/RafaSantos484/exercise-pose-trainer`.

The Kaggle repository `Yoga For All` is available at: `https://www.kaggle.com/datasets/jayasuryamarasani/yoga-for-all`.

The web app code is available in the repository: `https://github.com/RafaSantos484/ml-exercise-validator/`. It can be tested on any device with a web browser at: `https://ml-exercise-validator.vercel.app/`.

# References


Bazarevsky, V., Grishchenko, I., Raveendran, K., Zhu, T., Zhang, F., and Grundmann, M. (2020). Blazepose: On-device real-time body pose tracking.

Dias, V., Mauricio, C., and Peres, F. (2023). Análise e orientação de postura nos exercícios de calistenia usando estimativa de pose humana. In *Anais do XX Congresso Latino-Americano de Software Livre e Tecnologias Abertas*, pages 154–157, Porto Alegre, RS, Brasil. SBC. DOI: https://doi.org/10.5753/latinoware.2023.236523.

Difini, G. M., Martins, M. G., and Barbosa, J. L. V. (2021). Human pose estimation for training assistance: a systematic literature review. In *Proceedings of the Brazilian Symposium on Multimedia and the Web*, WebMedia '21, page 189–196, New York, NY, USA. Association for Computing Machinery. DOI: https://doi.org/10.1145/3470482.3479633.

Draelos, R. (2019). Measuring performance: The confusion matrix. Blog post on Glass Box Medicine.

Fong, A. and Otto, M. (2025). Yoga for all. Kaggle Dataset, released under MIT License.

G, A., Anas, M., B, N. K., G, R., and Jituri, V. (2023). Ai fitness trainer using human pose estimation. *International Journal of Engineering Research & Technology (IJERT)*, 11(08). RTCSIT – 2023.

Gonçalves, J., Palhares, J., Soares, V. N. G. J., and Neves, P. A. (2024). Detection and pose adjustment in physical exercises using computer vision techniques: Approaches, challenges and opportunities. *Revista de Informática Teórica e Aplicada*, 31(1):11–31. DOI: https://doi.org/10.22456/2175-2745.135436.

Han, J., Kamber, M., and Pei, J. (2011). *Data Mining: Concepts and Techniques*. Morgan Kaufmann, San Francisco, CA, 3 edition.

Ko, Y.-M., Nasridinov, A., and Park, S.-H. (2024). Real-time ai posture correction for powerlifting exercises using yolov5 and mediapipe. *IEEE Access*, 12:195830–195853. DOI: https://doi.org/10.1109/ACCESS.2024.3516723.

Kwon, Y. and Kim, D. (2022). Real-time workout posture correction using opencv and mediapipe. *The Journal of Korean Institute of Information Technology*, 20:199–208. DOI: https://doi.org/10.14801/jkiit.2022.20.1.199.

Martinez, J. J. L., Rodríguez-Roiz, J. M., and Cánovas, C. S. (2020). Musculoskeletal injuries secondary to exercise during confinement by the pandemic covid-19. *Medicina Clínica (English Edition)*, 155(5):221–222. DOI: https://doi.org/10.1016/j.medcle.2020.05.013.

Moreira, R., Teixeira, S., Fialho, R., Miranda, A., Lima, L. D. B., Carvalho, M. B., Alves, A. B., Bastos, V. H. V., and Teles, A. S. (2024). Validity analysis of monocular human pose estimation models interfaced with a mobile application for assessing upper limb range of motion. *Sensors*, 24(24). DOI: https://doi.org/10.3390/s24247983.

Newcombe, R., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A., Kohli, P., Shotton, J., Hodges, S., and Fitzgibbon, A. (2011). Kinectfusion: Real-time dense surface mapping and tracking. pages 127–136. DOI: https://doi.org/10.1109/ISMAR.2011.6092378.

Peffers, K., Tuunanen, T., Rothenberger, M., and Chatterjee, S. (2007). A design science research methodology for information systems research. *Journal of Management Information Systems*, 24:45–77.

Pires, J., Oliveira, M., Fonseca, B., Ribeiro, M., Giglio, B., and Calzado, A. (2025). Ergonomic assessment using human pose estimation: A real-time approach with yolo and blazepose. In *Anais Estendidos do XXV Simpósio Brasileiro de Computação Aplicada à Saúde*, pages 293–298, Porto Alegre, RS, Brasil. SBC. DOI: https://doi.org/10.5753/sbcas_estendido.2025.7623.

Rocha, J. Q. S., da Silva, L. S., Pintanel Freitas, M., Mendes Delpino, F., Rombaldi, A. J., de Almeida Paz, I., Schröeder, N., Santos Feter, J., Nascimento da Silva, C., Leal da Cunha, L., Cassuriaga, J., Feter, N., Cozzensa da Silva, M., Pereira Vieira, Y., Lucia Caputo, E., and Fossati Reichert, F. (2024). The use of digital platforms and physical activity practice in a population from southern brazil: Findings from the pampa cohort. *Preventive Medicine Reports*, 44:102816. DOI: https://doi.org/10.1016/j.pmedr.2024.102816.

Roggio, F., Trovato, B., Sortino, M., and Musumeci, G.

(2024). A comprehensive analysis of the machine learning pose estimation models used in human movement and posture analyses: A narrative review. *Heliyon*, 10(21):e39977. DOI: https://doi.org/10.1016/j.heliyon.2024.e39977.

Schlegel, K., Jiang, L., and Ni, H. (2024). Using joint angles based on the international biomechanical standards for human action recognition and related tasks.

Sugawara, E. K. (2022). Estimativa de postura: Uma abordagem multicâmera para classificação e avaliação de exercícios físicos. Master's thesis, Universidade Federal de Santa Catarina.

Xu, H., Bazavan, E. G., Zanfir, A., Freeman, B., Sukthankar, R., and Sminchisescu, C. (2020). Ghum ghuml: Generative 3d human shape and articulated pose models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (Oral)*, pages 6184–6193.

Zheng, C., Wu, W., Chen, C., Yang, T., Zhu, S., Shen, J., Kehtarnavaz, N., and Shah, M. (2023). Deep learning-based human pose estimation: A survey. *ACM Computing Surveys*, 56(1):1–37. DOI: https://doi.org/10.1145/3603618.