Computer Vision Based on a Modular Neural Network for Automatic Assessment of Physical Therapy Rehabilitation Activities

João A. Francisco and Paulo Sérgio Rodrigues

***Abstract*— Physical rehabilitation techniques during the treatment of clinical pathology are one of the most challenging areas for the medical structure, patients, and families. In large and continental countries, remote monitoring of this treatment is essential. However, equipment and medical follow-up during exercises still have high costs. With the improvement of computer vision and machine learning techniques, some computational, less expensive alternatives have been proposed in the literature. However, monitoring patients during physical rehabilitation exercises with the help of artificial intelligence by a health professional, especially from the capture of visual signals, is still a challenge and poorly explored in the scientific-technological literature. This work aims to propose a new methodology based on computer vision and machine learning for remote tracking of the body joints of patients during physiotherapy rehabilitation exercises. As a new contribution, this work presents a modular neural network architecture composed of two modules: one for detecting physical exercises and another for measuring how much is correct. Another contribution is a strategy for expanding databases, considering that generic databases for this type of exercise are rare on the internet. The results showed that both modules obtained more than 90% of accuracy in recognition and their respective validation.**

***Index Terms*— Modular neural network, OpenPose, physical therapy.**

I. Introduction

THE development of machine learning combined with Artificial Intelligence is mainly due to the vast application areas, with engineering, biology, and healthcare as great examples [4]. Despite undeniable advances, a critical issue for the importance of Artificial Intelligence [19] is the use of low-cost technologies for the operation of predictive models, such as standard cameras and IMU (Inertial measurement unit) sensors for methodologies that use digital signals as input on the network.

The health area is one of the fields that most benefits from the use of artificial intelligence, such as cancer treatment with Convolutional Neural Networks [21]; the treatment of chronic diseases such as diabetes [6]; the prevention of falls [16]; the rehabilitation of patients who had a stroke [14], to name a few examples. The results of such studies are promising, and the accuracy of the models can be comparable to treatments performed by health professionals. Nowadays, there is a hybrid treatment of patients in which the professional uses methodologies that employ machine learning to assist in the procedure and follow-up of the patient [14].

In recent times, with the advancement of artificial intelligence, cheaper sensors and promising results in the literature have accomplished a new field of study for treating people who had a stroke, especially in recognizing rehabilitative exercises. Using cameras in an indoor environment or sensors attached to the patient's body, It is now possible to monitor rehabilitative movements more precisely [3]. Under these technologies, healthcare professionals can efficiently monitor their patients with increasing quantitative and qualitative analysis. This technology allows patients to perform their treatment entirely remotely [18].

However, there are still many challenges in rehabilitation using Artificial Intelligence (AI). Many methodologies do not consider the severity of the patient's pathology. In contrast, the assessment by the AI models does not distinguish between a patient with a severe or mild stroke, which makes the quantitative and qualitative evaluation relatively poor. The lack of disease specificity can be important information to the health professional during the patient's treatment. There are also challenges when the healthcare professional wants to evaluate each step of the rehabilitative activity. One challenge occurs when the AI model shows the moment when the exercise is performed incorrectly and what could be done to improve the rehabilitation movement.

The work proposed here presents two main contributions. The first is a modular neural network for recognizing and assessing the assertiveness of physical therapy activities. The other is data augmentation of the databases used in the first contribution. As far as we know, neither of the

Manuscript received 5 May 2022; revised 2 November 2022; accepted 27 November 2022. Date of publication 2 December 2022; date of current version 5 May 2023. *(Corresponding author: Paulo Sérgio Rodrigues.)*

João A. Francisco Jr. is with the Electrical Engineering Department, Centro Universitário FEI, São Bernardo do Campo, São Paulo 09850-901, Brazil (e-mail: joao_junior174@outlook.com).

Paulo Sérgio Rodrigues is with the Computer Science Department, Centro Universitário FEI, São Bernardo do Campo, São Paulo 09850-901, Brazil (e-mail: psergio@fei.edu.br).

Digital Object Identifier 10.1109/TNSRE.2022.3226459

two contributions has ever been presented in the related literature.

As a future impact, these two contributions allow the construction of intelligent computational systems to aid distance physical therapy and further studies of therapies related to previous pathologies whose treatment indicated the aforementioned physical therapy exercises.

II. Fundamentals of the Proposed Methodology

The general idea of the proposed methodology is to use popular videos available on the internet of people performing three types of AROM-type (Active Range of Motion) rehabilitative physical exercises: squat, hip extension, and knee flexion. After capturing the exercises, the body joint's identification and validation are carried out. Then, to identify the body joints of the person performing the exercises, an AI-based library called Openpose is used. The Openpose is a well-known library developed by [5], which uses neural network concepts to identify 25 human body joints. It is commonly used in the literature for gesture recognition.

Four angles, studied as sufficient to describe the exercises, were extracted: from the armpits, hip, knee, and lower limbs, [13], [15], [20]. These angles were stored in two action banks. Finally, to validate the exercises, two modules with a back-propagation network were built, the Detection Module and the Measure Module. Therefore, this section explains the concepts used to develop the methodology further presented in Section III. The concepts presented here are: AROM-type exercises; Openpose; Action Banks; and Modular Neural Networks.

A. AROM-Type Exercises

The AROM (Active Range Motion) is a category of therapeutic exercises where the health professional measures the distance in which a joint of the human body can be moved from two points. These exercises help keep the joints flexible, reduce pain and improve balance and strength. Before starting treatment, the physical therapist usually measures the patient's range of motion [8]. Three types of exercises were selected for this article: squat, knee flexion, and hip extension. Each of these exercises has specific angles to consider. Physical therapists determine such angles in empirical tests in clinics, as the work of [20] shows.

1) Squat Exercise: This exercise is accomplished by bending the knees to a specific angle and slightly bending the hips. The degree of knee flexion angle will determine the type of squat exercise, which is generally classified as mini-squat ($140° - 150°$), semi-squat ($120° - 140°$), half squat ($80° - 110°$) and deep squat ($<80°$). Such exercise is mainly performed to strengthen the lower parts of the body, including the thighs, hips, and buttocks. It is usually performed with bare hands, called body-weight squats, or it can be performed with the help of weights and dumbbells [20]. The Fig. 1(a) show this type of exercise.

2) Knee Flexion Exercises: These exercises have no predefined angles, which depend on the patient's dynamic and the severity of the pathology [15]. However, there are rules that the patient needs to follow for the exercise to be classified as knee flexion. Fig. 1(b) shows example angles that the exercise should consider.

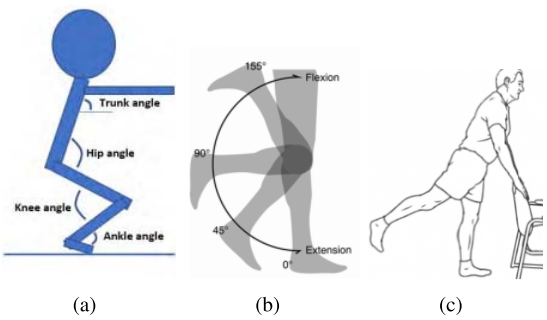

Fig. 1. Angle of each joint that characterizes a squat-type, knee flexion-type and hip extension-type exercises. Each angle has specific values to consider or are more flexible to changes [13], [15], [20].

3) Hip Extension: These exercises also have no predefined angle, depending on the patient's ability to perform the exercise correctly. The angle between the lower limbs must be close to 60° to perform correctly [13]. Fig. 1(c) demonstrates how the exercise should be performed.

B. Openpose

Openpose is an application proposed by [5] to recognize joints in the human body using AI. It is a real-time approach to detecting the body joints of multiple people in an image or video. The Openpose method uses a non-parametric representation, referred to as PAFs (Part Affinity Fields), to learn to associate human body joints with individuals in the image.

The network architecture includes several 7×7 convolutional layers preserving the receptive field. The output of each layer *kernels* is concatenated, following a similar approach to DenseNet [10]. The method has six steps: input image, trust maps; affinity maps; bipartite combination; and merging of the results. The algorithm detects 25 points of articulations that coordinate, specifying the pixel and the probability of that pixel corresponding to that articulation.

Our work uses the Openpose proposed by [5] as an initial method for the methodology proposed here.

C. Action Banks

The representation by Action Bank was first proposed by [17]. The objective was to propose a new representation of human action in the video. The action bank exploits a large set of action detectors that behave like the foundations of a high-dimensional action space that, combined with a simple linear classifier, can form the basis of a semantically rich representation for activity recognition. Fig. 2 illustrates the input images to form the Action Bank.

The action bank representation is a concatenation of max pooling features in the network used in the volumetric detection of input images. Each of these networks is called Detector. Fig. 3 shows the general taxonomy of the formation of the active bank. The SVM (Support Vector Machine) classifier detects action according to the feature vector.

Fig. 2. Exercises captured and grouped by class and frames used for training the networks [17].

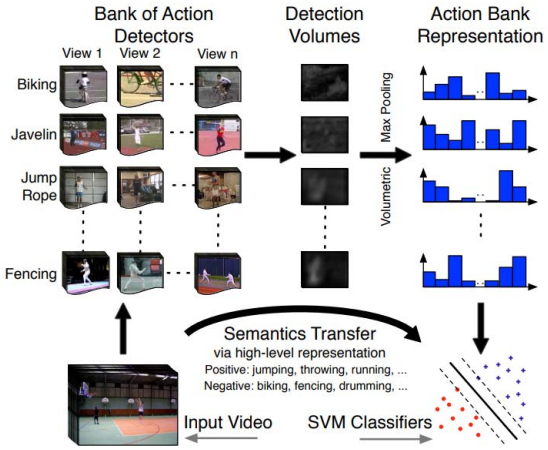

Fig. 3. General taxonomy of the formation of the action bank with max pooling and SVM classifier applied to features extracted [17].

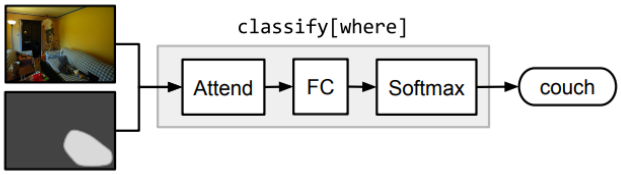

Fig. 4. Output image of the detection module taken from [1]. The entries are an attention mask and the original image. The type of object presented in the original is the output.

The Action Bank representation is used in this paper, but with some modifications. Instead of using the feature vector after max pooling, the Openpose feature vector will be used after extracting the kinematic data. Fig. 3 shows the blue part that the Openpose feature vectors will replace.

D. Modular Neural Networks

Modular neural networks separate large tasks into small sub-tasks representing a different neural network, called modules. This structure was first presented in the article by [1] and later extended to [2]. The main idea is to decompose big questions into linguistic substructures and dynamically instantiate modular networks with reusable components for recognizing objects. In the first paper by [1], all modules were trained together with five distinct modules: attention, re-attention, combination, detection, and measure.

The detection module receives an image and an attention mask. The output is the expected response of the object or required action. Fig. 4 shows an example of calling this module.

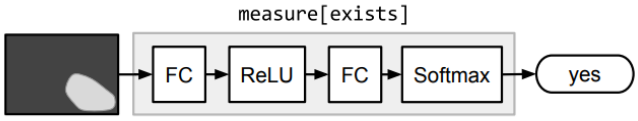

Fig. 5. Output image of the measure module taken from [1]. The input is an attention mask, and the output is the response of a boolean sentence as a question to the network.

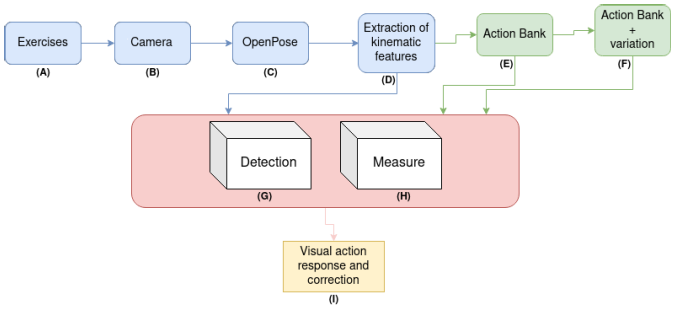

Fig. 6. Pipeline of the proposed methodology.

The measure module receives an attention mask and returns a distribution over labels. Fig. 5 shows an example.

III. The Proposed Methodology

This section aims to show the proposed methodology and the description of each module of Fig. 6. Each step is described in a specific subsection, and each subsection presents the purpose, input, and output for a given step.

A. General Explanation of Methodology

The methodology starts in step (A) when a patient performs a set of exercises for lower limbs, and a standard camera in step (B) captures their movements. In step (C), the Openpose (Section II-B) identifies the body's joints in each frame. In step (D), kinematic data are captured, like the joint's angles built in the previous step. In steps (D), (E), and (F), these kinematics data are stored in the action bank used for testing and training. In step (F), new movement variations were added to expand the actions of the previous step. This expansion allows the measure module to detect exercises performed incorrectly. This strategy is further described in more detail in Subsection III-A.6.

1) Step A: Exercises: In this step, the user performs exercises in an environment monitored by a standard camera. Patients with stroke may have poor mobility of the lower or upper limbs, depending on the severity [12]. If it is high severity, passive rehabilitative exercises are prescribed for the patient; that is, there is the help of the physical therapist. However, for patients who had little impact on limb movement, active exercises are prescribed in which the patient performs them alone [11]. Therefore, in this paper, active exercises were considered. The main objective is to evaluate how these exercises would be recognized in people who have some pathology and need to perform this set of activities.

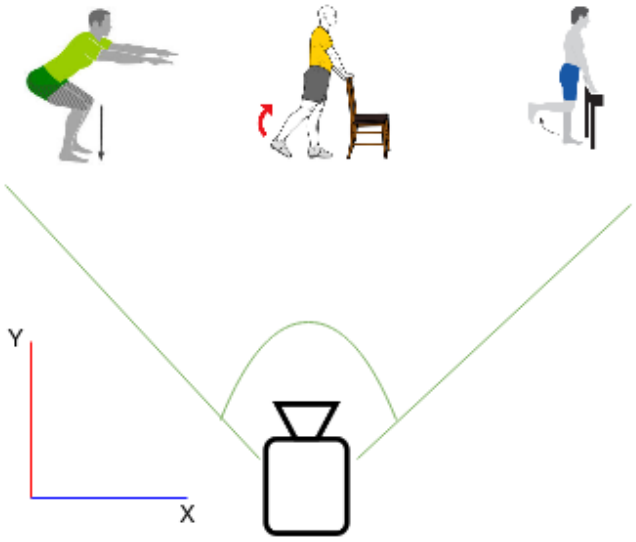

Fig. 7. Side view of the camera. All exercises must be performed orthogonally to the camera.

The exercises established for this work are of the type AROM, as explained in Section II-A. For selecting the three types of exercises, we follow the literature presented in Section II.

The first exercise is squat, as shown in Fig. 1(a). Knee angles were adopted as approximately 40°, as discussed in Section II-A with arms straight and extended forward and hips flexed over the knees. The second movement flexes the knee, as shown in Fig. 1(b). The knee angle was adopted to be approximate 100° to 120°. For the correct execution of this exercise, the lower limbs must perform the movements alternating with each other. The last movement is hip extension, as shown in Fig. 1(c). This exercise can be performed with or without support. The angle between the lower limbs can range from 50° to 100°.

2) Step B: Camera: Since the database used to train the networks presented here is built with popular databases found on the internet, the camera chosen for this step can be a simple standard camera. In this work, the user must perform the movements in the orthogonal perspective of the camera, as illustrated in Fig. 7.

3) Step C: OpenPose: This step aims to collect the frames captured from input camera and estimate the user's joints performing the exercise. Then, the OpenPose is used here. The program's output represents the body articulation of the person performing the activity in the captured frames. Furthermore, a matrix of the size of the frame is generated, where each position (i, j) corresponds to the probability of the point (i, j) in the frame belonging to a possible articulation. After filtering the pixels with the highest probabilistic value, a vector of 25 positions is obtained. Each (i, j) position stands for a 2D point containing the coordinates i and j of that pixel in the input image of the OpenPose.

4) Step D: Extraction of Kinematic Features: From the kinematic data, the angles between the joints are calculated. Fig. 8 shows a scheme for calculating angles according to

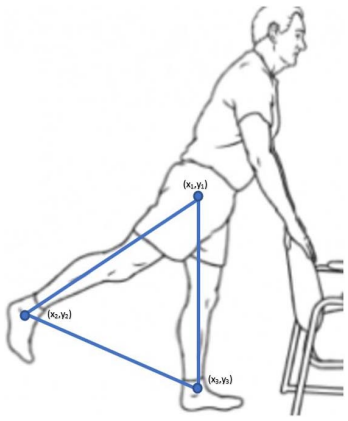

Fig. 8. Calculation of joint angle. Equation (1). The same type of triangulation is done for all other joints.

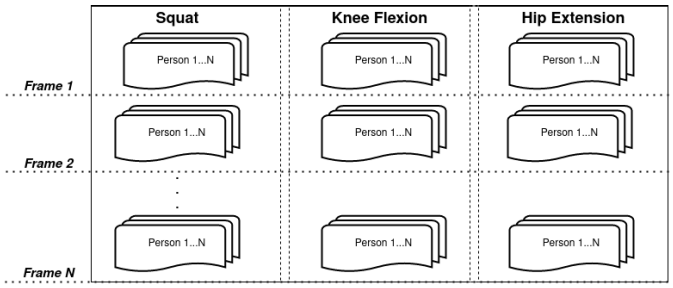

Fig. 9. Structure of the Action Bank.

the coordinates obtained in the previous step. Let D_3 be the Euclidean distance between points (x_1, y_1) and (x_3, y_3), D_2 the distance between (x_1, y_1) and (x_2, y_2) and D_1 a distance between (x_3, y_3) and (x_2, y_2). Then, the angle A is calculated using the Law of Cosines by Equation (1).

$$D_3^2 = D_2^2 + D_1^2 - 2.D_2.D_1.cosA \tag{1}$$

5) Step E: Action Bank: The action bank aims to store the vectors of angles obtained with the 25 points extracted from OpenPose. Fig. 9 illustrates the storage structure.

Each column of Fig. 9 shows the type of exercise a user performs. Each line stands for the moment at which the first frame was captured. After reaching an exercise state, the second frame is captured, until the last frame N, representing the end of the physical activity. Each matrix position in Fig. 9 shows an angle of the user's joints in a specific exercise. The present work uses publicly available databases of people undergoing physical therapy exercises.

The number of frames varies according to the video's size. After identifying the joints of a human body, only the angles are stored and used to train the detection and measure networks. Here, only four angles are considered and stored in each action bank position: armpit angle; knee; hip; and the angle between lower limbs, extracted using the strategy shown in Fig. 8. For example, a video containing 300 frames of a person performing squat exercises results in 300 vectors containing four angles each, which describe the type of exercise throughout the video. This data will be stored in a text file containing the four frames' four angles.

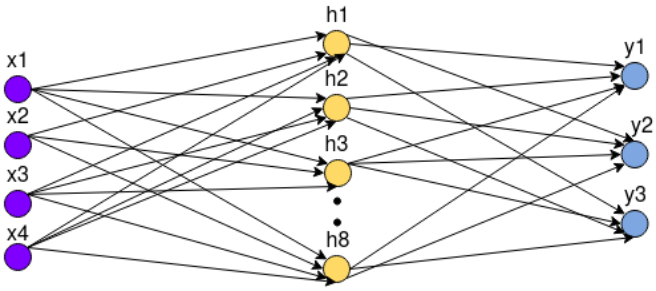

Fig. 10. Detection module model. The initial network has four input neurons, eight in the hidden layer with ReLU as the activation function and three in the output with sigmoid as the activation function.

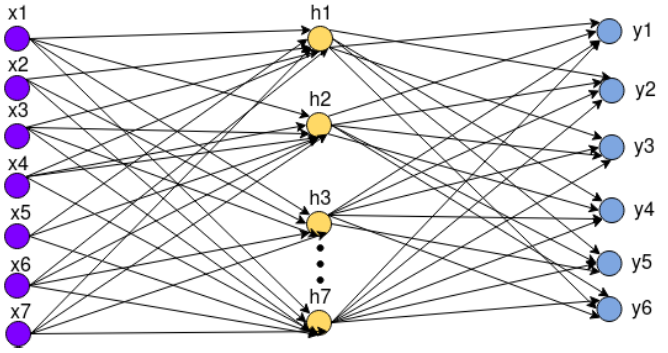

Fig. 11. Measure module. The initial network has seven input neurons, seven in the hidden layer with ReLU as the activation function and six in the output with sigmoid as the activation function.

6) Step F: Action Bank + Variation: This step stands for storing a copy of the action bank. This action bank aims to train the measure module, which is responsible for showing the region in which the exercise is being performed incorrectly. However, in order to expand the database and the assertiveness of the proposed architecture here, random noise was added to the angle values. More details about this expansion can be further seen in Section IV-A.1.

The network trained with this action bank outputs kinematic values regarding the angles of each user joint outside the ground truth, which are the data before adding the random variations. As far as we know, such an approach is not found in the literature, so this method is one of our contributions.

The angles considered incorrect are parameters further changed according to who uses the methodology and wants to classify it as an invalid exercise. In order to select the incorrect angles for this work, the angles are analyzed based on the Openpose method. Then, this work proposes the values of the parameters that would cause an incorrect exercise.

For the squat exercise, two joints are selected to change the angles of the arms and hips. For each N frame, angles between 123° and 170° are generated; squat-type exercises cannot be performed with the arms raised. For the hip, angles are generated between 170° and 180°, implying that the spine cannot be erect. In knee flexion, the angle of the lower limbs should be at most 27°. The hip extension exercise has several ways to be performed. Therefore, the exercise of this type is incorrect if it becomes knee flexion; that is, it is classified as knee flexion with the lower extremities at an angle above 27°.

7) Step G: Detection Module: This step aims to receive the angles of the exercises performed by a user and associate them with the corresponding class in the action bank. The model presented in Fig. 10 was proposed.

The module Input corresponds to the four angles that describe the proposed exercises. Initially, only one hidden layer contains eight neurons ($h1, h2, h3, \ldots, h8$). ReLU is the activation function of each hidden layer. As an output, there are three neurons, each representing a class of exercise. The activation function of the last layer was sigmoid. The weight of the network is initialized using the He method [9].

The input data to train this network comes from the action bank. The input network is a four-position vector, such as $[30, 30, 30, 60]$, representing the angles of the armpits, knee, hip, and lower limbs. The output is a three-position vector representing the type of exercise being performed. The vector $[0, 0, 1]$ represents the class of squat exercises, the vector $[0, 1, 0]$ the hip exercises, and the vector $[1, 0, 0]$ for knee flexion.

More hidden layers were added to verify which model was more accurate during training, testing, and generating the results. However, it is not enough to recognize all types of exercises based only on the action bank without variations. As in Step F (Section III-A.6), here in Step G, random noises were added to the angles to expand the database. More details about this expansion can be further seen in Section IV-A.1.

8) Step H: Measure Module: This step has the objective of assessing whether the exercise performed by the person is being performed correctly. If it is incorrect, it will show which member is in error. The model of this network can be seen in Fig. 11.

The network has seven input neurons, the first three being the network output of the detection module and the last four being the small random angles corresponding to the angles of the armpit, hip, knee, and lower limbs. For example, the vector [0,0,1,32,32,32,62], which describes a squat-type exercise with two degrees added as a variation, considering that the correct vector would be [0,0,1, 30,30,30,60].

There is also a hidden layer containing seven neurons, with ReLU as the activation function. The output layer also has the sigmoid activation function and six output neurons. The net weight was also initialized with the He method [9].

The input data to train this network of measure module comes from the action bank plus variation (Fig. 6 (H)). The output of this module is a six-dimensional vector. The first dimension of the output vector only indicates whether the exercise is right or wrong. The next five dimensions only indicate which type of error occurred in case the first dimension indicates that there was an error in the exercise performed by the user. Empirical tests were carried out to decide the number of hidden layers of each architecture (detection or measurement modules).

This network also changed the number of hidden layers during training, testing, and generating the results. Empirical tests were performed to determine the number of hidden layers of each architecture.

9) Step I: Visual Action Response and Correction: This step displays the results of the network output of the measure and detection modules. The results are presented in an image, superimposing in the captured video, the type of exercise, and the status. If it is being performed wrongly, a correction recommendation will appear.

IV. Results and Discussion

This section presents the experiment strategies and the results obtained after applying them. Therefore, this section is divided into two subsections: experimental strategy (Section IV-A) and obtained results (Section IV-B).

A. Experimental Strategy

The experimental strategy of this work is divided into five main steps: action bank database building, detection module training; measure module training; metrics; and hardware used. All are described in more detail in the following sections.

1) Action Bank Database Building: Section III presents the three classes of exercises: squat, knee flexion, and hip extension. On the other hand, this work used public videos of physical exercises to build the stock bank. Each database class has twelve videos containing between 100 and 500 frames, thus producing 36 videos with 3600 to 6000 frames. This first database is denoted here in our work as Base-Original.

The detection module recognizes the exercise even if it is incorrect. However, as all test examples are of people performing the exercises correctly, there is expected to be an error in the classification.

Also, a simple data-augmentation technique was created where small random oscillations were added to the Base-Original in the following body's articulation: armpit, hip, and lower limbs (legs). Thus, 36 new videos were obtained with oscillations in the angles, generating 72 videos: 36 without oscillations and 36 with oscillations. Therefore, the added random oscillations extrapolate these limits, creating new artificial videos and a second database denoted as Original-Oscillation.

Thus, an algorithm was developed to collect the exercises' original data and generate random angles in each frame. For the squat exercise, random angles were generated between 140° and 170° and between 30° and 50° for the hip. In the case of armpits, angles between 20° and 90° and between 90° and 140° were generated. For knee flexion, angles between 0° and 27° were generated in the lower limbs. For hip extension, angles between 20° and 90° and between 90° and 140° were generated for the armpits.

To create the training and testing database of the measure module, random oscillations with errors were added to the 36 original videos, following the strategy of Subsection III-A.6, thus creating a third database denoted here as Oscillation-Errors.

Thus, three databases were used for the training and testing phases: The Base-Original, with 36 original videos (used to train the detection module); The Original-Oscillation, with 72 videos (36 with random oscillation and 36 without such oscillation), also used to train the detection module; and Oscillation-Errors with 36 videos with random error oscillations. Thus, in the entire experiment, 108 videos were used.

2) Detection Module Training: As presented in Section III, the detection module is a back-propagation neural network containing three input neurons, initially with one hidden layer with eight neurons and three output neurons. This network was denoted as Network-Detection. In order to validate this network, the results were generated using the Base-Original and Original-Oscillation databases.

Nine architectures were generated to test the Base-Original and Original-Oscillation databases and validate which one obtains greater accuracy in the detection module. Setting epoch and batch size values: The first architecture has one hidden layer with eight neurons; the second architecture has one hidden layer with 12 neurons; the third architecture has two hidden layers; the fourth architecture has three hidden layers, and so on until the sixth architecture. The amounts of hidden layers were kept from the seventh to the ninth architectures, but this time only varying the epoch and batch size values. The number of neurons in the hidden layers from the third to the sixth architecture was empirically determined.

To perform tests, 10% of the data from both databases were considered (Base-Original and Original-Oscillation) for testing and 90% for training. It is expected that the accuracy of the architecture trained with the Original-Oscillation database will be higher when compared to the accuracy of the architecture obtained with the Base-Original database, making it possible to recognize the exercises being performed incorrectly

3) Measure Module Training: As described in Section III, the measure module is a back-propagation neural network containing seven input neurons, initially one hidden layer with eight and six output neurons. This network was denoted as the Network-Measure. In order to validate this architecture, results were generated using the Oscillation-Errors database.

As in the Network-Detection training, the Network-Measure also varied the parameters: number of hidden layers, epochs, and batch size. The total of generated architectures was also nine. Under constant epoch and batch size values, the first architecture has one hidden layer, the second architecture has two hidden layers, the third architecture has three hidden layers, and so on until the eighth architecture. The amounts of hidden layers were maintained in the eighth and ninth architectures, again varying the epoch and batch size values.

After the training step, we performed tests with videos that make up 10% of the selected database to validate the Network-Measure with the Oscillation-Errors database.

4) Metrics Used to Validate the Results of the Detection and Measure Modules: Two metrics were used to validate the accuracy of the detection and measure modules: confusion matrix and area under the ROC curve. The best architecture with Original-Oscillation database was used for training, and the best architecture with the Oscillation-Errors database was used for the measure module. Six confusion matrices were generated for each class of the Network-Detection and six others for each Network-Measure class.

5) Hardware Used to Perform the Experiments: The Python language was used to develop the entire methodology. The library Keras, Numpy, Scikit-Learn, and Tensorflow were used

TABLE I
NINE ARCHITECTURES TRAINED WITH BOTH BASE-ORIGINAL AND ORIGINAL-OSCILLATION DATABASES

Architectures	Neurons by Hidden Layer	Epochs	Batch size
1-1	8	50	5
2-1	10	50	5
3-1	10;8	50	5
4-1	10;8;10	50	5
5-1	10;8;10;10	50	5
6-1	10;8;10;10;8	50	5
7-1	10;8;10;10;8	50	8
8-1	10;8;10;10;8	100	8
9-1	10;8;10;10;8	250	8

to build the two modules. In order to run the Openpose program, the version compiled for Windows was selected. The computer used to run Openpose, train, and test modules have the following specifications: AMD Ryzen 5 3600 CPU; B450 AORUS M motherboard; Dedicated RTX 2060 GPU; 16 GB of memory and 256 GB of SSD. The code in python and videos of this article can be found in the GitHub repository.[1]

B. Results

1) Detection Module: The first test was accomplished with the Base-Original and the Original-Oscillation databases. Thus, nine architectures were developed, with 90% of data used for training and 10% for testing in each database.

The nine types of architectures are summarized in Table I. The first column represents the types of built architecture. There are nine architectures, so they were numbered 1-1 through 9-1. The Neurons per Hidden Layer column indicates, for each architecture, the number of hidden layers and the number of neurons in each one. For example: in architecture 1-1, there is a hidden layer with eight neurons; architecture 4-1 has three hidden layers, with respectively 10, 8, and 10 neurons in each. The Epochs column contains the learning iterations and the Batch Size, the amount of data trained per iteration.

In architecture 2-1, the number of neurons in the first hidden layer was 10. The results were satisfactory, but new layers were added from architecture 3-1 to architecture 6-1, varying the number of neurons between 10 and 8 to improve the prediction. After architecture 6-1, it was noticed that adding more hidden layers did not improve the results, generating many false positives. Therefore, the number of epochs and batch size were changed. In turn, architecture 9-1 achieved satisfactory accuracy before over-fitting. The same strategy was used to train the architectures with the Original-Oscillation database.

The accuracy of the architectures trained with the Base-Original and Original-Oscillation databases are shown in Table II. Accuracies of architecture 9-1 are similar for both databases. Observing the graphs shown in Fig. 12 (training with the Base-Original database, blue line with circles, and training with the Original-Oscillation database, red line with asterisks), we can note that the architectures, with each change, tended to increase the value of their accuracy.

Architecture 9-1 for both databases obtained the best results, besides being similar. Thus, it was taken to be the default

[1]https://github.com/joaoJunior174/mestrado

TABLE II
ACCURACY OBTAINED AFTER TRAINING THE NINE ARCHITECTURES WITH TWO TYPES OF DATABASES: BASE-ORIGINAL AND ORIGINAL-OSCILLATION

Database/ Architecture	1-1	2-1	3-1	4-1	5-1	6-1	7-1	8-1	9-1
Base-Original	79,04%	82,34%	81,56%	84,88%	87,27%	86,66%	82,82%	90,23%	92,11%
Original-Oscillation	61,26%	67,97%	78,82%	78,52%	84,40%	80,19%	82,33%	88,56%	90,07%

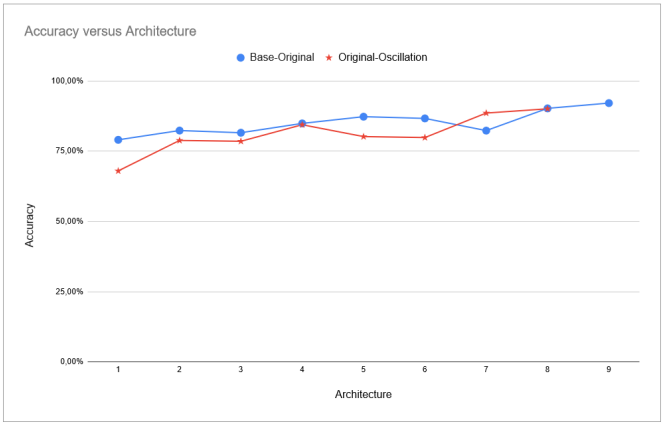

Fig. 12. Accuracy obtained for architectures 1-1 to 9-1 after training the networks using the Base-Original database (blue line with circles); Accuracy obtained for architectures 1-1 to 9-1 after training the networks using the Original-Oscillation database (Red line with asterisks).

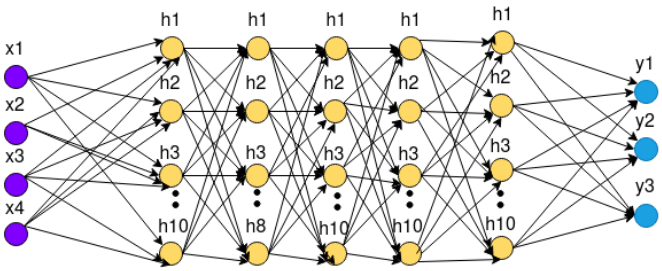

Fig. 13. Final model of the detection module: the network has five hidden layers with ReLU as the activation function and three neurons in the output with sigmoid as the activation function.

architecture of the detection module. Fig. 13 presents the network model of this architecture.

When comparing the accuracy of architecture 9-1 for Base-Original and Original-Oscillation in Table II, we can note that the results for Base-Original are superior. However, even with a decrease in accuracy, the results of architecture 9-1 after training its network model with the Original-Oscillation database are more acceptable, as the network identified the type of exercise even with small oscillations. The image of Fig. 14 shows the result of the detection module with architecture 9-1 after training the network with the Base-Original.

In the upper left corner, the classification of the exercise as knee flexion is incorrect when it should be a squat. The reason is that all the training examples are of people performing the exercise correctly. Thus, if there is an example of someone

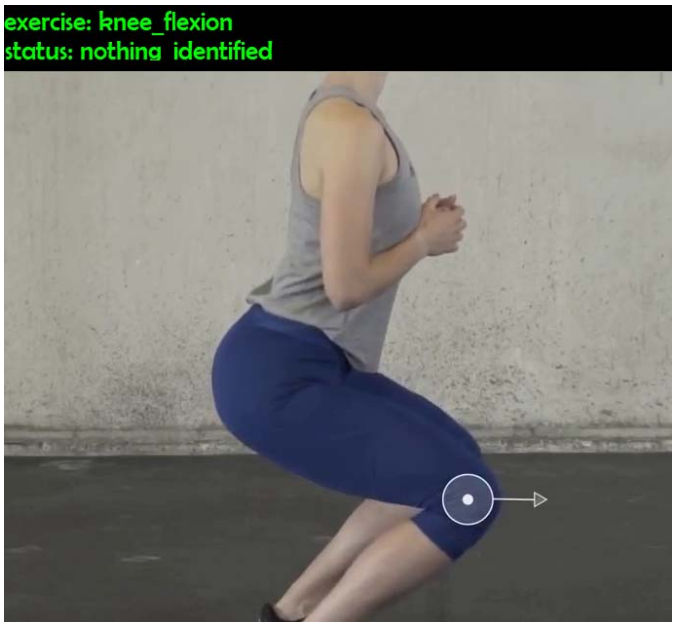

Fig. 14. Squat exercise detection failed with architecture 9-1 after training the network with Base-Original.

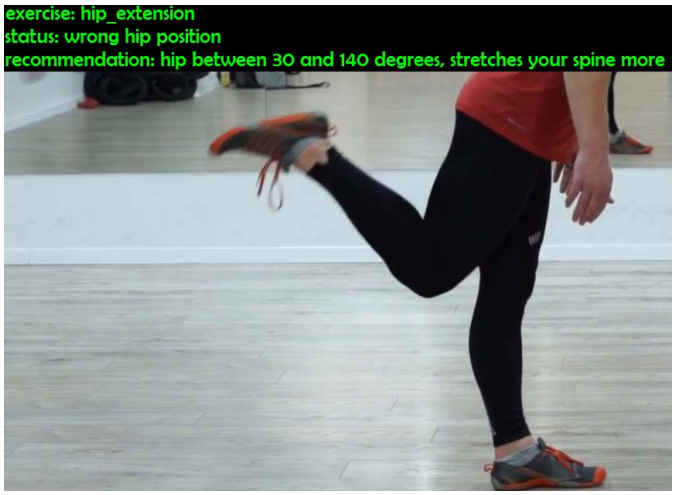

Fig. 15. Failure to detect knee flexion exercise.

doing the exercise incorrectly but still within the exercise class, the classification may present not coherent results.

The same occurred for an exercise of another class, as shown in the image of Fig. 15. Also, the upper left corner indicates the classification as a hip extension when it should be knee flexion.

However, new acceptable results are generated when training the architecture 9-1 network with the Original-Oscillation database. The image in Fig. 16 shows the result of the squat exercise, indicating in the upper left corner it is incorrect. Although its execution is incorrect, the function of the detection module uniquely identifies only the exercise type. The same occurs for the knee flexion exercise, as shown in the image of Fig. 17. Also, in the upper left corner, we can see the correct detection of the exercise.

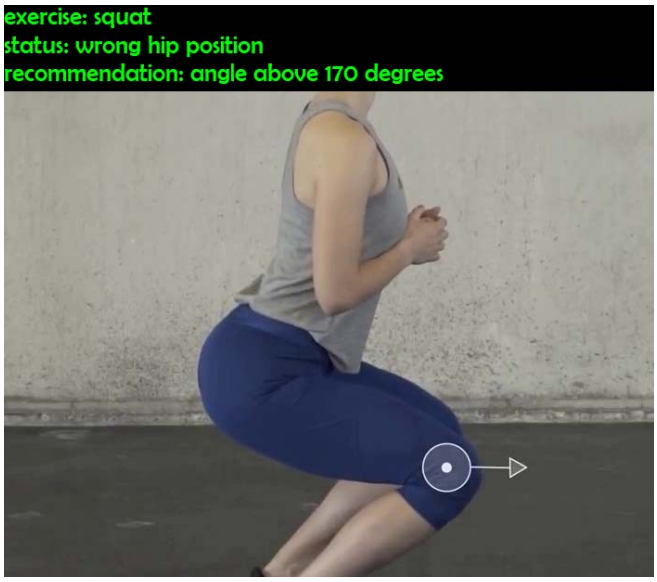

Fig. 16. Correct detection of the squat exercise.

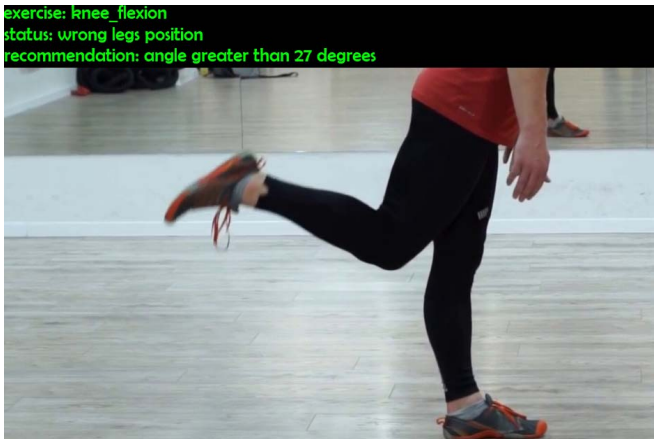

Fig. 17. Correct detection of knee flexion exercise.

Fig. 18. Correct detection of knee flexion exercise.

More results were generated to validate the accuracy of architecture 9-1 with the Original-Oscillation database. The images in Fig. 18 and 19 show these results.

2) Measure Module: Nine new architectures were built to test the measure module due to the following factors:

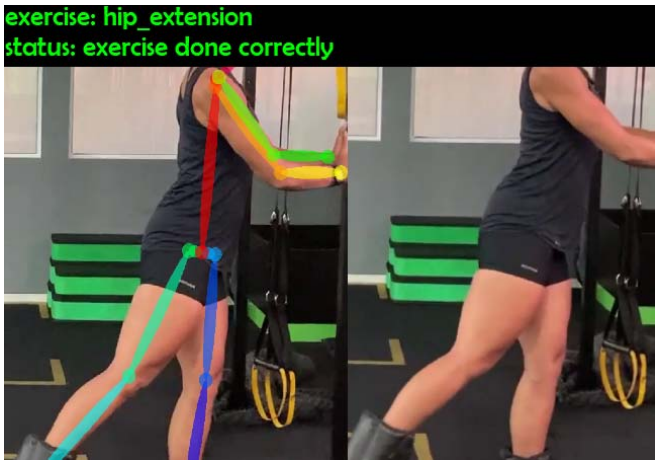

Fig. 19. Correct detection of hip extension exercise.

TABLE III
NINE ARCHITECTURES TRAINED WITH THE OSCILLATION-ERRORS DATABASE

Architecture	Neurons by Hidden Layer	Epoch	Batch size	Accuracy
1-2	7	50	5	80, 20%
2-2	10	50	5	91, 64%
3-2	10;12	50	5	94, 02%
4-2	10;12;10	50	5	84, 61%
5-2	10;12;10;8;10	50	5	93, 77%
6-2	10;12;10;8;10;12;12	50	5	91, 93%
7-2	10;12;10;8;10;12;12	50	5	94, 30%
8-2	10;12;10;8;10;12;12;8	50	5	93, 59%
9-2	10;12;10;8;10;12;12;8	120	5	94, 54%

- The number of input neurons changed;
- The type of exercise performed also represents the angles with oscillations;
- The output is five neurons that describe the joints;

Thus, both input and output influence the input architecture.

The Oscillation-Errors database was used to train the networks of the new architectures. Table III was generated to summarize the nine types of architectures and their respective accuracy.

The strategy to build these new architectures was the same for the detection module. Then, layers were randomly added to each architecture, varying the number of neurons in each hidden layer. The best result was obtained with architecture 9-2. The graph in Fig. 20 summarizes the accuracy according to the presented architectures. Architecture 9-2 achieved the best results. Therefore, it was chosen to be the default architecture of the measure module. Fig. 21 presents the network model of this architecture.

More results were generated to validate the accuracy of architecture 9-2 with the Oscillation-Errors database. The images in Fig. 22 and 23 show these results.

3) Confusion Matrix and ROC Curve: Based on the confusion matrix and the ROC curve, the following considerations were accomplished: the used threshold, 0.1, is in the range of [0, 0.5]; in the networks of the detection and measure modules, the cut-off threshold was 0.5; the area under the curve was calculated; since all graphs showed the same pattern, the formula side multiplied by height, in this case, specificity multiplied by sensitivity, was used; For the detection module, the architecture 9-1 was used with its network trained with the Original-Oscillation database; and for the measure module, architecture 9-2 was used with its network trained with the Oscillation-Errors database.

As shown in Table IV, all classes with architectures 9-1 and 9-2 presented satisfactory results. As explained in [7], the

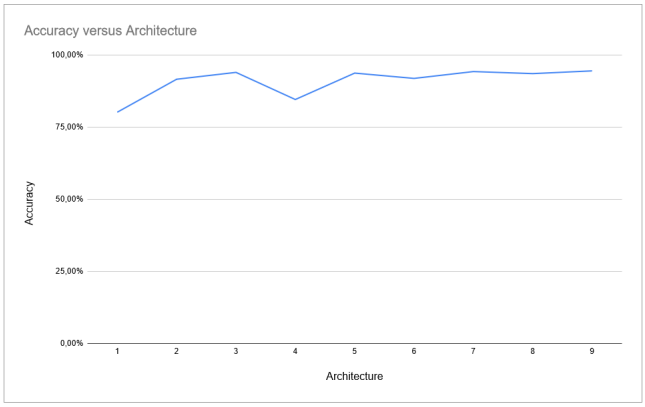

Fig. 20. Accuracy obtained for architectures 1-2 to 9-2, after training the networks using the Oscillation-Errors database.

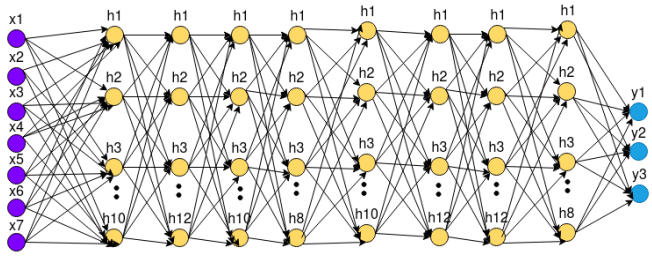

Fig. 21. Final model of the measure module. The network has eight hidden layers, with all hidden layers with ReLU as the activation function and three neurons in the output with sigmoid as the activation function.

Fig. 22. Correct detection of knee flexion exercise.

TABLE IV
AREAS UNDER THE CURVE FROM CONFUSION MATRICES OF EACH TRAINING CLASS IN BOTH MODULES (DETECTION AND MEASURE)

Module	Training Class	Architecture	Area under the curve
Detection	Squat Exercise	9-1	0, 9
Detection	Hip Extension	9-1	0, 7
Detection	Knee Flexion	9-1	0, 85
Measure	Squat Exercise with incorrect arms position	9-2	0, 8
Measure	Squat Exercise with incorrect hip position	9-2	0, 9
Measure	Squat Exercise with incorrect arms and hip positions	9-2	0, 9
Measure	Knee Flexion with incorrect legs position	9-2	1
Measure	Exercise done Correctly	9-2	0, 9

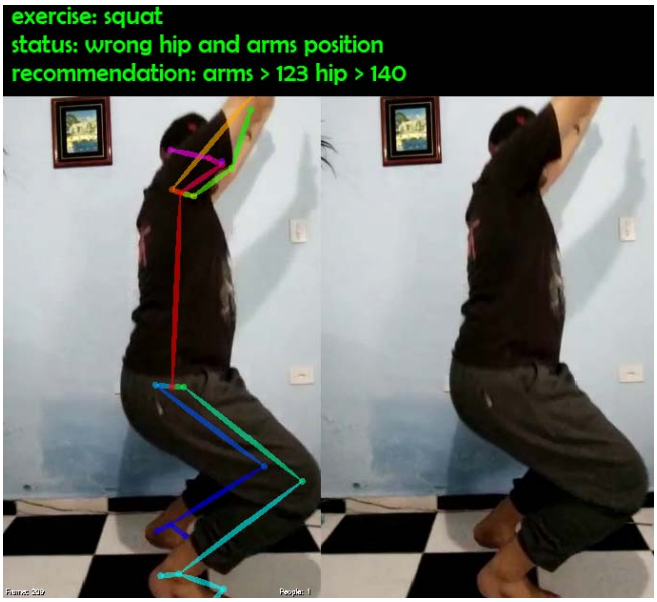

Fig. 23. Correct detection of squat exercise.

closer the area under the curve is to 1 (100% assertiveness), the more assertive the network. This leads architectures 9-1 and 9-2 to have approximately 90% accuracy.

V. Conclusion

This work contributes to the scientific literature on recognizing and evaluating rehabilitative physical exercises, capturing videos of people performing physical activities with a standard RGB camera. A nine-step methodology is proposed to capture videos of a person performing rehabilitative activities such as a squat, hip extension, and knee flexion. Two neural networks, called detection and measure modules, were developed. Both modules are responsible for recognizing and validating the type of exercise. The final step is responsible for presenting the results in the text on the video.

Based on the results, we can note that the accuracies for architectures 9-1 and 9-2, trained respectively with Original-Oscillation and Oscillation-Errors, were approximately 90%. It is possible to notice that the area under the curve of all classes comes close to the ground truth. With the areas under the curve being close to the ground truth, it is evident that the architecture 9-1 and 9-2 networks are classifiers with satisfactory accuracy.

The proposed methodology can help in two lines of research. First, it can deal with extracting information from physical therapy exercises. Then, it is possible to further statistical analysis, allowing the construction of artificial agents that can assist health professionals in reaching conclusions regarding diagnoses and treatments. The second line can be the construction of artificial agents that accompany the patients and help them in real-time to execute the exercises correctly.

Acknowledgment

The authors would like to thank the CNPq and CAPES, the Brazilian agencies for Scientific Financing, FAPESP (Sao Paulo Research Foundation), as well as to FEI (Ignacian Educational Foundation) a Brazilian Jesuit Faculty of Science Computing and Engineering for the support of this work.

References

[1] J. Andreas, M. Rohrbach, T. Darrell, and D. Klein, "Deep compositional question answering with neural module networks," 2015, *arXiv:1511.02799*.

[2] J. Andreas, M. Rohrbach, T. Darrell, and D. Klein, "Learning to compose neural networks for question answering," in *Proc. HLT-NAACL*, 2016, pp. 1–10.

[3] F. Ayoubi, S. Chamouni, O. Zein, and A. R. Sarraj, "Virtual reality movement therapy for post-stroke upper limb rehabilitation trial," in *Proc. 5th Int. Conf. Adv. Biomed. Eng. (ICABME)*, Oct. 2019, pp. 1–3.

[4] J. P. R. Caicedo, J. Verrelst, J. Muñoz-Marí, J. Moreno, and G. Camps-Valls, "Toward a semiautomatic machine learning retrieval of biophysical parameters," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 7, no. 4, pp. 1249–1259, Apr. 2014.

[5] Z. Cao, G. Hidalgo, T. Simon, S. E. Wei, and Y. Sheikh, "OpenPose: Realtime multi-person 2D pose estimation using part affinity fields," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 1, pp. 172–186, Jan. 2021.

[6] H.-C. Chan, J.-C. Chen, S.-W. Chien, Y.-F. Chen, and C.-T. Bau, "Evaluation of intelligent system to the control of diabetes," in *Proc. Int. Symp. Comput., Consum. Control*, Jun. 2012, pp. 585–588.

[7] G. A. Diamond, "ROC steady: A receiver operating characteristic curve that is invariant relative to selection bias," *Med. Decis. Making*, vol. 7, no. 4, pp. 238–243, Dec. 1987.

[8] L. A. Elrefaei, B. Azan, S. Hakami, and S. Melebari, "JCAVE: A 3D interactive game to assist home physiotherapy rehabilitation," *Int. J. Multimedia Appl.*, vol. 11, no. 2, pp. 1–20, Apr. 2019.

[9] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1026–1034.

[10] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4700–4708.

[11] I. Indrawati, K. Sudiana, and M. Sajidin, "Active, passive, and active-assistive range of motion (ROM) exercise to improve muscle strength in post stroke clients: A systematic review," in *Proc. 9th Int. Nursing Conf.* SciTePress, 2019, pp. 329–337.

[12] W. Ling, G. Yu, and Z. Li, "Lower limb exercise rehabilitation assessment based on artificial intelligence and medical big data," *IEEE Access*, vol. 7, pp. 126787–126798, 2019.

[13] D. A. Neumann, "Kinesiology of the hip: A focus on muscular actions," *J. Orthopaedic Sports Phys. Therapy*, vol. 40, no. 2, pp. 82–94, Feb. 2010.

[14] N. Norouzi-Gheidari, M. F. Levin, J. Fung, and P. Archambault, "Interactive virtual reality game-based rehabilitation for stroke patients," in *Proc. Int. Conf. Virtual Rehabil. (ICVR)*, Aug. 2013, pp. 220–221.

[15] Y. Qi, C. Boon Soh, E. Gunawan, K.-S. Low, and A. Maskooki, "Measurement of knee flexion/extension angle using wearable UWB radios," in *Proc. 35th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. (EMBC)*, Jul. 2013, pp. 7213–7216.

[16] J. P. Queralta, T. N. Gia, H. Tenhunen, and T. Westerlund, "Edge-AI in LoRa-based health monitoring: Fall detection system with fog computing and LSTM recurrent neural networks," in *Proc. 42nd Int. Conf. Telecommun. Signal Process. (TSP)*, Jul. 2019, pp. 601–604.

[17] S. Sadanand and J. Jason Corso, "Action bank: A high-level representation of activity in video," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 1234–1241.

[18] Q. Sanders, V. Chan, R. Augsburger, S. C. Cramer, D. J. Reinkensmeyer, and A. H. Do, "Feasibility of wearable sensing for in-home finger rehabilitation early after stroke," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 28, no. 6, pp. 1363–1372, Jun. 2020.

[19] W. Zhang et al., "Distributed deep learning strategies for automatic speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2019, pp. 5706–5710.

[20] M. A. Zulkifley, N. A. Mohamed, and N. H. Zulkifley, "Squat angle assessment through tracking body movements," *IEEE Access*, vol. 7, pp. 48635–48644, 2019.

[21] M. Saric, M. Russo, M. Stella, and M. Sikora, "CNN-based method for lung cancer detection in whole slide histopathology images," in *Proc. 4th Int. Conf. Smart Sustain. Technol. (SpliTech)*, Jun. 2019, pp. 1–4.