An augmented system based on machine learning for Boccia assisted gameplay

João Cruz [1], Vinícius Silva[2], João Sena Esteves [1,2], Filomena Soares [1,2]

[1] Department of Industrial Electronics, School of Engineering, University of Minho, Guimarães, Portugal
[2] R&D Centre Algoritmi, Guimarães, Portugal
(a85939, a65312)@alunos.uminho.pt, sena@dei.uminho.pt, fsoares@dei.uminho.pt

Abstract. In order to promote the practice of sports, several approaches using technology have been employed to gamify and augment the user experience. Following this trend, the research group proposed an approach to encourage the practice of Boccia, while promoting social inclusion and reducing the amount of time it takes for newcomers to the sport to become proficient by gaining knowledge of game tactics. The present work focus on the detection, in real-time, of Boccia gestures for the framework proposed in a previous work by using a wearable device to detect the gestures. To evaluate the correct functioning of the system, several types of tests were carried out. First, the developed machine learning model was evaluated in terms of accuracy, recall, among others. Then, the gesture detection system was tested with 15 participants that executed the different Boccia gestures while using the wearable placed on the wrist. Finally, tests were carried out to integrate the gesture detection module into the framework proposed in a previous work. The tests yielded positive results that allowed the validation of the use of the system in the Boccia game.

Keywords: Activity monitoring; Boccia; Gesture Recognition; Machine Learning; Wearable

1 Introduction

The practice of sports, being fundamental, has been neglected. A large part of the national adult population is not physically active, which translates into a higher risk of non-communicable diseases and mortality [1]. The benefits of practicing sports are not only physical, but they are also a catalyst for personal development, which is even more expressive in people with disabilities [2]. The significance of social development in the life of individuals with disabilities becomes apparent considering the previous remarks. As a result, it is crucial to formulate and execute strategies aimed at mitigating the effects of disability-induced social exclusion and fostering inclusivity, which can be achieved through engagement in sports.

Boccia is a precision ball sport that has gained attention for its ability to promote inclusivity. The sport requires a combination of physical and mental skills, including

This is an Author Accepted Manuscript version of the following chapter: Cruz, J., Silva, V., Esteves, J.S., Soares, F., An Augmented System Based on Machine Learning for Boccia Assisted Gameplay, published in Advances in Production. ISPEM 2023. Lecture Notes in Networks and Systems, vol 790, edited by Burduk, A., Batako, A., Machado, J., Wyczółkowski, R., Antosz, K., Gola, A., 2023, Springer, reproduced with permission of Springer Nature Switzerland AG. The final authenticated version is available online at: http://dx.doi.org/10.1007/978-3-031-45021-1_20

hand-eye coordination, strategic planning, and decision-making. It can be played individually or in teams, and the goal is to throw or roll a set of colored balls as close as possible to a target ball, known as the jack. A system using augmented reality to enhance the Boccia game experience, aiding novice players in comprehending key tactical and physical aspects of the sport, ultimately resulting in a more enjoyable learning experience was proposed by the research team in [3]. One of the features of the proposed framework concerns the detection of the key gestures that can be executed by the players during a Boccia game using the data from a wearable device.

Several authors have proposed different approaches for the recognition of gestures using data from wearable devices [4]–[8]. Traditional approaches combine the use of hand crafted features with machine learning algorithms in order to detect the gestures [6], [7]. In recent times, deep learning methods have gained popularity and shown exceptional performance in several image recognition tasks. In fact, some studies have employed these techniques, specifically Convolution Neural Networks (CNNs), to address the gesture recognition problem, [9]–[11]. Several studies demonstrate that CNNs outperform traditional hand-crafted approaches as they can learn intricate motion features, [12]–[14]. Nonetheless, conventional approaches of gesture recognition using deep learning methods typically employ the use of full RGB images, which could result in an increase in the complexity of the model due to the large dimension of the input features. This results in more complex and slower models, less practical for solving real-time large-scale problems. In order to tackle the dimensionality problem and computation complexity, some authors proposed the use of skeleton based representation data obtained from a 2D camera in order to recognise gestures, [9], [10], [13]. This process consists in obtaining the 2D position of the skeleton joints from RGB images and then convert it into an image representation. This data is then fed as input to the deep learning model.

Following this trend, the present work focus on the developments concerning the gestures recognition module of the framework developed by the research team and presented in [3]. In order to fulfill this goal, a dataset was created and a deep learning model based on a Convolution Neural Network (CNN) was trained. The data from the wearable sensor is converted into an image representation. The final model runs in real-time on a CPU.

The following paper is organized in 5 sections. The proposed system and the developed approach to the gesture recognition process are described in Section 2. The results are presented in Section 3 and discussed in section 4. Final remarks and future work are addressed in Section 5.

2 Proposed System

The proposed system consists of a wearable device M5StickC [15], a computer, and a camera placed directly above the Boccia court (Fig. 1). The view from the camera is used as input of a deep learning model in order to detect, in real-time, the Boccia balls within the camera Field-of-View (FOV) and sort them according to color (red, blue, or white), which was the focus of previous work [3], [16]. To compute the distance of each red and blue ball and the jack (white ball), the corresponding centroid coordinates of each ball within the camera FOV are used, allowing to map all balls and to compute the score for the current game situation.

For each move there is an associated throwing movement that may facilitate the task of placing the ball in the desired position. There are two main gestures that are usually executed by the players, the upper throwing and the under throwing gestures (Fig. 2). The present work focus on the development of a deep learning model to detect these two gestures in real-time.

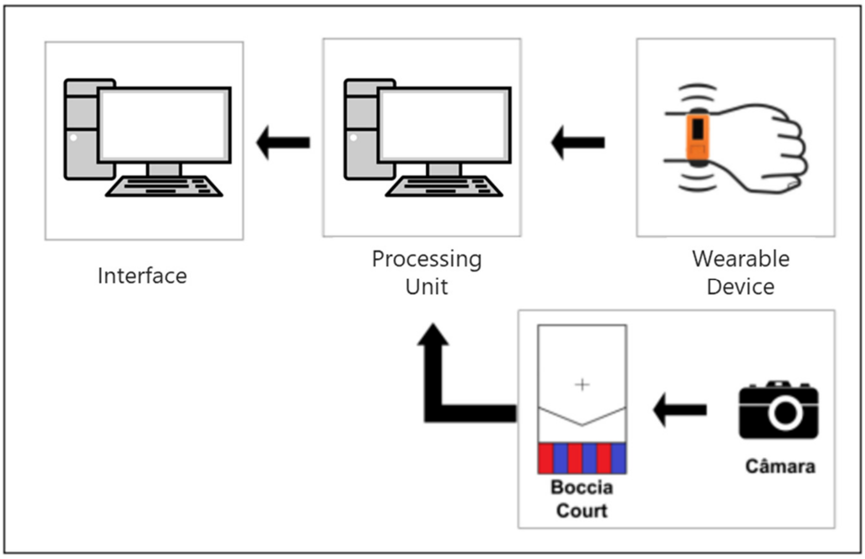


Fig. 1. Proposed system framework.

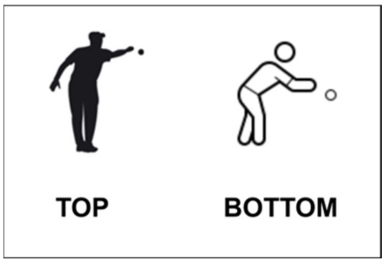


Fig. 2. Main gestures usually executed by Boccia players: on the left the upper throwing gesture and on the right the under throwing gesture.

2.1 Dataset

To detect movements, a set of data is collected from the M5StickC development board. These data are all coming from a 6-axis Inertial Measurement Unit (IMU) in the MPU6886. Three values are collected for each of the accelerometer and gyroscope axes, as well as the corresponding roll, pitch and yaw values.

In order to obtain data for training the deep learning model, the development board was attached to a bracelet. This bracelet was placed on the right wrist of the participants. Ten samples of data referring to each type of movement (Fig. 3) that is intended to be detected were captured. Therefore, each participant performed 10 throws over, 10 throws under and remained at rest for 10 times as well.

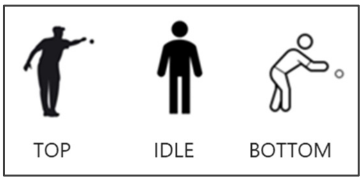


Fig. 3. Types of movements considered in the construction of the dataset.

Data capture was performed by twelve adult participants, which resulted in a total of 360 data files (12 participants, 3 types of recorded movements, 10 times each). The age group of the participants was between 18 and 80 years old and individuals of both sexes were invited. In each of those 360 files, the accelerometer and gyroscope values are arranged on the x, y and z axes, as well as the roll, pitch and yaw values during a time interval of two seconds of movement.

2.2 Data Processing

The first step is the normalization of the data values between 0 and 255 to integrate into color pixel values. Once normalized and in order to augment the amount of data to be provided to the neural network, extra handcrafted features are computed on the incoming data from the wearable device. The extra features computed are the cumulative moving average and the RMS values, both made for all the x, y and z components of the accelerometer and gyro, as well as the roll, pitch and yaw.
The wearable placed on the player's wrist provides the data frame, as seen in (1).

$$\text{gyroX/gyroY/gyroZ/accX/accY/accZ/roll/pitch/yaw} \tag{1}$$

The data obtained over several frames are used as input for the action recognition model to classify the action. All generated images have a 64 x 12 x 3 size, where 64 is the number of data samples per movement, 12 parameters from the wearable representing the height of the image, and 3 is the number of channels, Fig. 4.
The first three lines of the image are the gyroscope data, accelerometer data and RPY (roll-pitch-yaw) data. Each pixel has the data variable X as the red color component, the data variable Y as the green color component, and the data variable Z as the blue color component.

For the following three lines, the cumulative moving average values were used. The pixel assignment is very similar to the previous lines, but now the color components are matched to the MMA (cumulative moving average) for all the color components (x, y and z) of the received data.
For line number seven, eight and nine the reasoning is similar again. In this case, the assignment is made according to the RMS values of each data component.
In the last three lines, the pixel values are assigned according to the data components. Line ten was associated with all data components in X, line eleven with all data components in Y and finally, line twelve with all data components in Z. The detailed constitution of each pixel can be analyzed in Fig. 4. Fig. 5 shows some samples of the generated images per class.

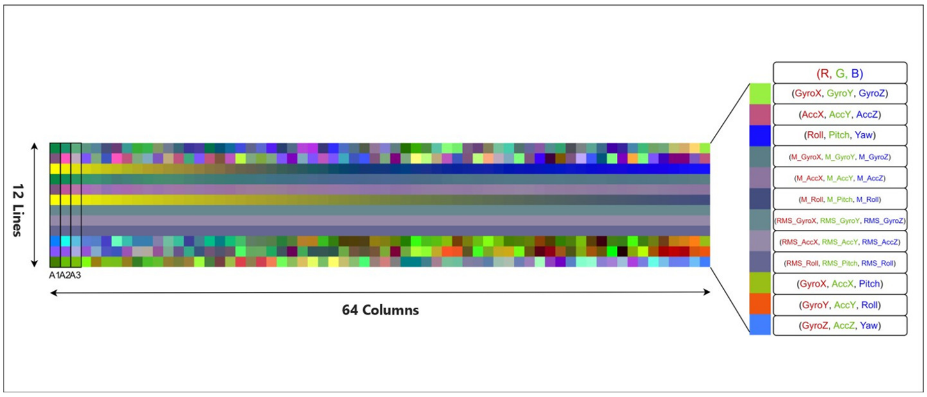


Fig. 4. Generated image-based representation from the incoming data from the wearable device.

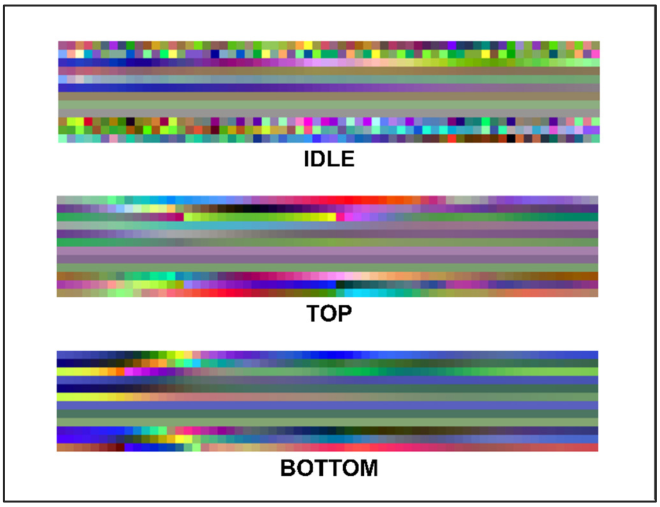


Fig. 5. Samples of the generated images per-class.

2.3 Developed model

The model follows a CNN architecture and consists of a total of six convolutional layers each with a 3x3 kernel, a stride value of 1, see Fig. 6. It was decided to use groups of two 3x3 filters in a stack since these are less demanding from a computational point of view when compared to the use of only one filter of larger dimensions [9]. Thus, three groups of two stacked convolutional layers are left, where a batch normalization layer is added to each group. The ReLu function is used as an activation function.

A max polling layer with a size of 2x2 is added after the second and fourth convolutional layers. This layer's main function is to subsample the feature maps, maps generated by convolutions. As in this work the layer has a size of 2x2 the spatial resolution of the data will be halved. After the last convolutional layer, a global averaging pooling layer is used to replace the fully connected layers, since these layers can constitute an overfitting problem, [17]. To finalize the model, a fully connected layer was added to handle the classification part. The entire model was trained in TensorFlow and the processing capabilities of the google COLAB tool were used, since it allows the use of a dedicated GPU, in order to train the model more rapidly.

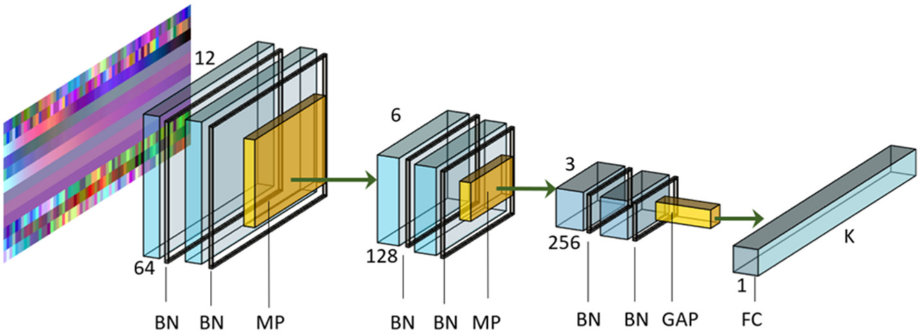


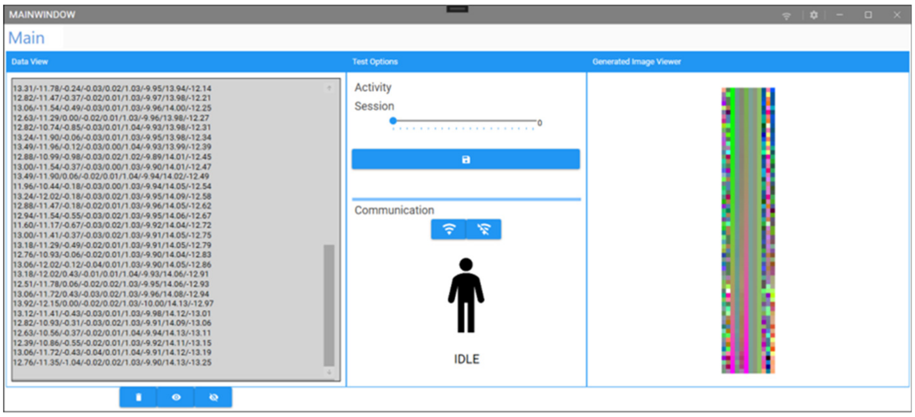
Fig. 6. The proposed deep learning model architecture.

The trained model was then imported to the developed Graphical User Interface (GUI).

2.4 GUI

In order to extract the data from the wearable device and to validate the trained deep learning model, a GUI was developed as seen in Fig.

The GUI consists of five main areas. The data view area on the left side is where the user can watch the data flow coming from the wearable wristband. Then the generated image area, on the right side, where is possible to analyze the image that feeds the CNN model. In the centre, the activity area is used to store session data, and the communication area is responsible for establishing the connection between the wearable and the PC. Finally, the detection area located at the bottom of the centre area of the interface is where the predicted class of movement is displayed to the user via image.



```
13.31/-11.78/-0.24/-0.03/0.02/1.03/-9.95/13.94/-12.14
12.82/-11.47/-0.37/-0.02/0.01/1.03/-9.97/13.98/-12.21
13.06/-11.54/-0.49/-0.03/0.01/1.03/-9.96/14.00/-12.25
12.63/-11.29/0.00/-0.02/0.01/1.03/-9.96/13.98/-12.27
12.82/-10.74/-0.85/-0.03/0.01/1.04/-9.93/13.98/-12.31
13.24/-11.90/-0.06/-0.03/0.01/1.03/-9.95/13.98/-12.34
13.49/-11.96/-0.12/-0.03/0.00/1.04/-9.93/13.99/-12.39
12.88/-10.99/-0.98/-0.03/0.02/1.02/-9.89/14.01/-12.45
13.00/-11.54/-0.37/-0.03/0.00/1.03/-9.90/14.01/-12.47
13.49/-11.90/0.06/-0.02/0.01/1.04/-9.94/14.02/-12.49
11.96/-10.44/-0.18/-0.03/0.00/1.03/-9.94/14.05/-12.54
13.24/-12.02/-0.18/-0.03/0.02/1.03/-9.95/14.09/-12.58
12.88/-11.47/-0.18/-0.02/0.01/1.03/-9.96/14.05/-12.62
12.94/-11.54/-0.55/-0.03/0.02/1.03/-9.95/14.06/-12.67
11.60/-11.17/-0.67/-0.03/0.02/1.03/-9.92/14.04/-12.72
13.00/-11.41/-0.37/-0.03/0.02/1.03/-9.91/14.05/-12.75
13.18/-11.29/-0.49/-0.02/0.01/1.03/-9.91/14.05/-12.79
12.76/-10.93/-0.06/-0.02/0.01/1.03/-9.90/14.04/-12.83
13.06/-12.02/-0.12/-0.04/0.01/1.03/-9.90/14.05/-12.86
13.18/-12.02/0.43/-0.01/0.01/1.04/-9.93/14.06/-12.91
12.51/-11.78/0.06/-0.02/0.02/1.03/-9.95/14.06/-12.93
13.06/-11.72/0.43/-0.03/0.02/1.03/-9.96/14.08/-12.94
13.92/-12.15/0.00/-0.02/0.02/1.03/-10.00/14.13/-12.97
13.12/-11.41/-0.43/-0.03/0.01/1.03/-9.98/14.12/-13.01
12.82/-10.93/-0.31/-0.03/0.02/1.03/-9.91/14.09/-13.06
12.63/-10.56/-0.37/-0.02/0.01/1.04/-9.94/14.13/-13.11
12.39/-10.86/-0.55/-0.02/0.01/1.03/-9.92/14.11/-13.15
13.06/-11.72/-0.43/-0.04/0.01/1.04/-9.91/14.12/-13.19
12.76/-11.35/-1.04/-0.02/0.02/1.03/-9.90/14.13/-13.25
```

Fig. 7. Developed GUI used to collect the data and to validate the trained deep learning model.

As previously mentioned, this project follows the work done by the research team in [3], and so the GUI Fig. 8 was developed in the scope of the same project.
The GUI allows to monitor in real-time the Boccia game, by detecting the balls within the FOV of the camera. With the collected information it was possible to develop a system that calculates the score of the game and that tells the player the best tactical play for that moment.

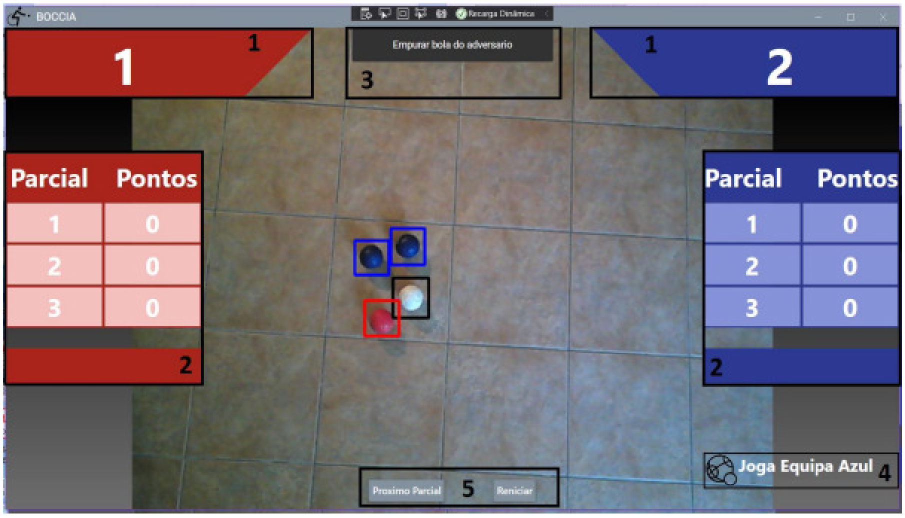


Fig. 8. The GUI developed by the research team in [3].

Figure 9 shows the update version of the GUI, where it was added an area that shows to the user the prediction of the deep neural network model regarding the type of gesture recognized.

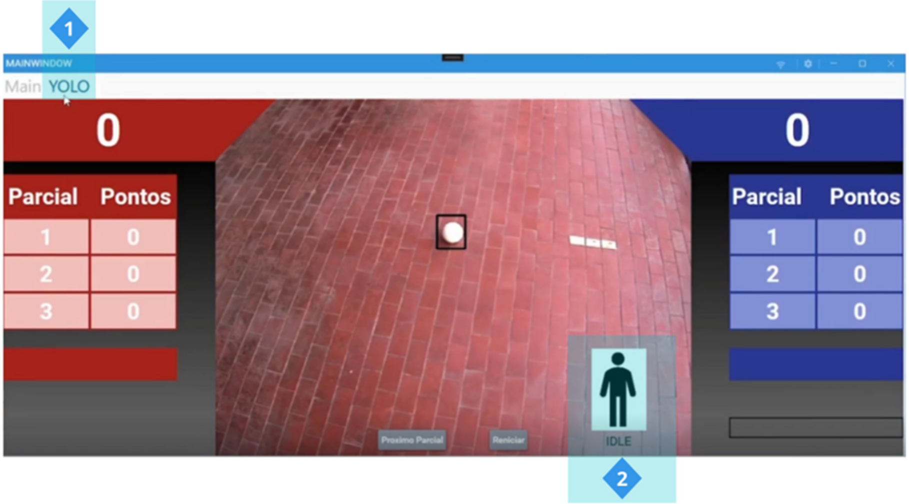


Fig. 9. The updated GUI with the prediction of the deep neural network model regarding the type of gesture recognized (2).

3 Results

This section presents the results obtained throughout the development of the proposed system. First, the performance of the deep learning model is evaluated through metrics. Afterwards, a pilot test was conducted in order to assess, the recognition accuracy of the deep leaning model in a real-world environment. Finally, the results of the integration of the gesture recognition subsystem in the system previously developed by the research group are addressed.

3.1 Model Results train and testing

The deep learning model was tested to evaluate its performance. The results of the full analysis – model accuracy, precision, recall, F1-Score, and confusion matrix are presented below.

Table 1 shows the overall performance of the gesture recognition model with an accuracy of 97.0%.

Table 2 presents the values of precision, recall and an F1-Score for each class of movements. For the under throw, the model achieved a precision of 100.0%, a recall and F1-Score of 95.0% and 98.0% respectively. Secondly, for the idle class the model achieved a precision, recall and, an F1-Score of 93.0%. Finally, for the upper throw the model achieved a precision of 96.0%, a recall value of 100.0% and an F1-Score of 98.0%.

The confusion matrix for the three classes of gestures can be observed in Fig. 10. More specifically, the under throw with a classification rate of 100%. The classification rate for the idle class and for the upper throw was 93.0% and 96.0%, respectively.

Table 1. Overall performance of the gesture recognition model.

Metric	Value
Accuracy	97%

Table 2. Performance of the model per-class.

	Precision	Recall	F1-Score
Bottom	1.00	0.95	0.98
Idle	0.93	0.93	0.93
Top	0.96	1.00	0.98

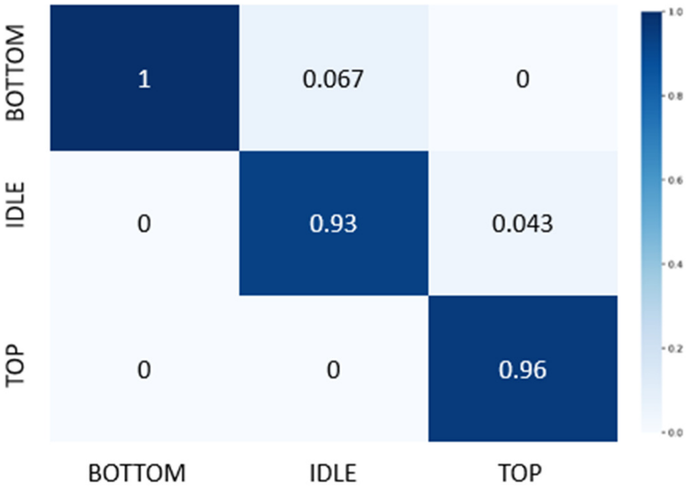


Fig. 10. Trained model confusion matrix.

3.2 Pilot test results

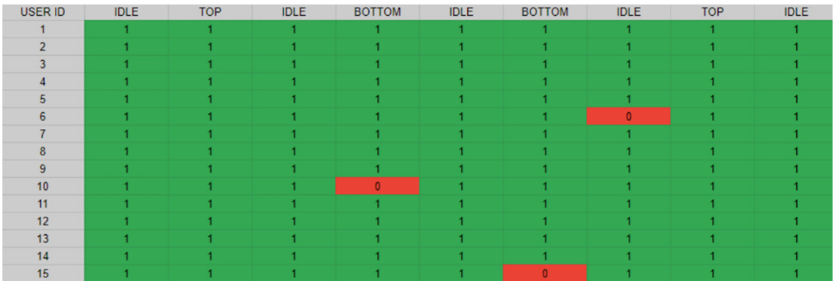
A pilot test was carried out to test the gesture detection system in real-time. The participants considered for the test were 15 adults aged 18 to 81 years old, with 26.7% of the participants being female and 73.3% being male. The experimental unit consisted of a processing unit (PC) and a wearable wristband equipped with the M5StickC. With the bracelet placed on the wrist of the subject, the researcher asked for the execution of the following sequence of gestures. (IDLE - TOP - IDLE - BOTTOM - IDLE - BOTTOM - IDLE - TOP – IDLE). The sequence was chosen so that both the upper throw and the

under throw gestures were performed the same number of times (twice each). The sequence also ensured that there would always be a gap between each throw where the individuals would be idle between gestures. During the execution of the tests, all the results were noted down by a supervisor.
Of the data collected, the Fig. 11 was created. The cells represented in green are the set of gestures correctly predicted in real-time by the artificial intelligence model and the cells in red represent the gestures where the system failed to predict.

Table 3 shows the overall performance of the real-time test results with an accuracy of 98.0%.

Table 4 presents the values of precision, recall and an F1-Score for each class of movements. For the under throw, the model achieved a precision of 100.0%, a recall and F1-Score of 93.0% and 97.0% respectively. Secondly, for the idle class the model achieved a precision of 97.0%, recall of 99.0% and, an F1-Score of 98.0%. Finally, for the upper throw the model achieved a precision of 97.0%, a recall value of 100.0% and an F1-Score of 99.0%.

The confusion matrix for the three classes of gestures can be observed in Fig. 12. More specifically, the under throw with a classification rate of 100%. The classification rate for the idle class and for the upper throw was 97.4% and 96.8% respectively.



USER ID	IDLE	TOP	IDLE	BOTTOM	IDLE	BOTTOM	IDLE	TOP	IDLE
1	1	1	1	1	1	1	1	1	1
2	1	1	1	1	1	1	1	1	1
3	1	1	1	1	1	1	1	1	1
4	1	1	1	1	1	1	1	1	1
5	1	1	1	1	1	1	1	1	1
6	1	1	1	1	1	1	0	1	1
7	1	1	1	1	1	1	1	1	1
8	1	1	1	1	1	1	1	1	1
9	1	1	1	1	1	1	1	1	1
10	1	1	1	0	1	1	1	1	1
11	1	1	1	1	1	1	1	1	1
12	1	1	1	1	1	1	1	1	1
13	1	1	1	1	1	1	1	1	1
14	1	1	1	1	1	1	1	1	1
15	1	1	1	1	1	0	1	1	1

Fig. 11. Results of the experiment conducted

Table 3. Model test performance obtained from the tests conducted.

Metric	Value
Accuracy	98.0%

Table 4. Model test performance per-class obtained from the experiment conducted.

	Precision	Recall	F1-Score
Bottom	1.00	0.93	0.97
Idle	0.97	0.99	0.98
Top	0.97	1.00	0.99

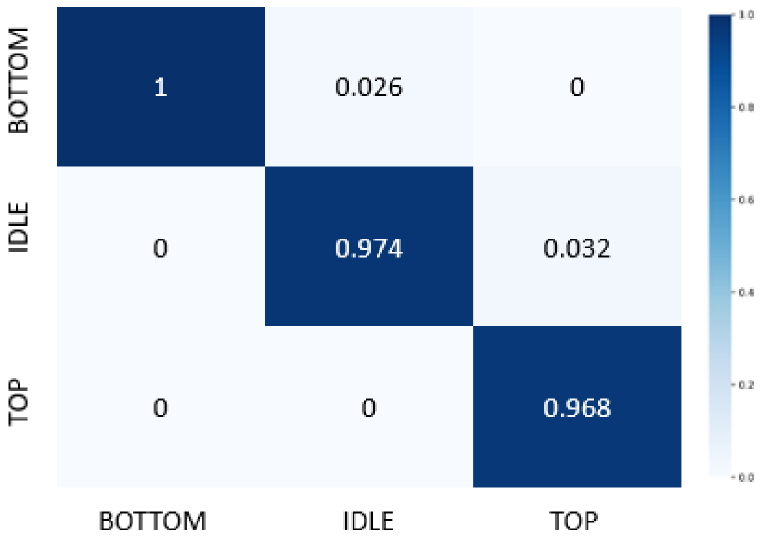


Fig. 12. Model test confusion matrix.

4 Discussion

Starting by analyzing the confusion matrix of the artificial intelligence model (Fig. 10), it is possible to conclude that all three classes had a precision over 90.0% (diagonal values). However, the model predicted 6.7% of cases as being idle when they were, in fact, an under throw movement. The model also predicted 4.3% of cases as being part of the upper throw class despite being, in fact, part of the idle class. In general, the results of the model performance-wise are positive since gesture recognition is performed correctly in most of the times.

Regarding the performance of the real-time results and analyzing the confusion matrix (Fig. 12), the under-throw class presents a 100.0% precision value. Concerning the upper throw class, there is a precision of 97.4% and finally, in the idle class a precision of 96.8%. However, the model also predicted 3.2% of cases as being upper throw despite being part of the idle class and 2.6% of the cases as being part of the idle class when they were, in fact, an under throw. The developed system had an accuracy value of 98.0%

By comparing the confusion matrix of the model (Fig. 10) and the confusion matrix obtained experimentally in the real-time tests carried out (Fig. 12) is possible to get some conclusions. It's possible to observe that in the under-throw class, the results in terms of precision are the same in both situations, that the real-time tests and the idle class presented results slightly better than model results. Finally, in the upper throw class, the real-time tests also had a smaller error than the model prediction.

The computational time was computed considering the average execution time of the model. The tests ran on a notebook equipped with an Intel(R) Core (TM) i7-8650 quad-core CPU with 16 GB of RAM. The inference is done by using OpenCV integrated Deep Neural Network module that allows to import saved TensorFlow models [18]. Based on the performed tests, the average execution time of the model is 9 milliseconds, which ensures the real-time capability of the developed approach.

5 Final Remarks and Future Work

Sports play a vital role in promoting physical and mental health, social inclusion, and overall well-being. Boccia has gained a significant attention, as it allows to promote the practice of sports and to foster inclusivity. In recent years, Boccia has been further enhanced by technological advancements with the goal to gamify and augment the user experience while promoting the practice of sports.

Following this trend, the research team proposed in [3] an augmented reality framework to facilitate learning and improve the player's experience while playing Boccia. One of the modules of the framework consists in detecting the user throwing gestures. The present work focus on the development of this module. In order to detect gestures, a deep learning model based on a CNN architecture was developed and trained. In order to train the model, data was collected from various throws executed by several participants. The data was collected through a wearable placed on the participants' wrist, integrated into a bracelet.

Regarding the results of the developed model, a precision value of 93% was obtained for the still class, 100% for the underhand launch class, and 96% for the overhand throw class. Concerning the real-time test, the following results were achieved: a precision value of 97.4% for the still class, 100% for the underhand throw class, and 96.8% for the overhand throw class. The developed approach was integrated in the final GUI. The model execution time is, on average, 9 milliseconds.

Future work includes a continuous improvement of the presented approach by, for example, augmenting the dataset and implementing other deep learning architectures in order to improve the model performance. Additionally, experiments will be conducted in order to test the system proposed in [3] with the model developed in the present work.

Acknowledgments

The authors would like to thank the volunteers who helped test the developed system. This work has been supported by national funds through FCT – *Fundação para a Ciência e Tecnologia* within the R&D Units Project Scope: UIDB/00319/2020.

References

1. World Health Organization, “WORLD REPORT ON DISABILITY,” 2011.
2. A. Burchell, “The Importance of Sport to the Disabled,” The Commonwealth Health Minister’s Book, 2006.
3. A. Calado, S. Marcutti, V. Silva, G. Vercelli, P. Novais, and F. Soares, “Towards a virtual coach for boccia: Developing a virtual augmented interaction based on a boccia simulator,”

in VISIGRAPP 2020 - Proceedings of the 15th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications, 2020. doi: 10.5220/0009142602170224.

4. D. Weinland, R. Ronfard, and E. Boyer, “A survey of vision-based methods for action representation, segmentation and recognition,” Computer Vision and Image Understanding, vol. 115, no. 2, pp. 224–241, 2011, doi: 10.1016/j.cviu.2010.10.002.
5. S. Amendola, L. Bianchi, and G. Marrocco, “Movement Detection of Human Body Segments: Passive radio-frequency identification and machine-learning technologies.,” IEEE Antennas Propag Mag, vol. 57, no. 3, pp. 23–37, 2015, doi: 10.1109/MAP.2015.2437274.
6. D. Wilson and A. Wilson, “Gesture Recognition Using The XWand,” 2004. [Online]. Available: http://www.ri.cmu.edu/pub_files/pub4/wilson_daniel_h_2004_1/wilson_daniel_h_2004_1.pdf%5Cnhttp://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.1.2780
7. J. Wu, G. Pan, D. Zhang, and G. Qi, “Gesture recognition with a 3-d accelerometer,” Proceedings of the 6th International Conference on Ubiquitous Intelligence and Computing, vol. 5585, pp. 25–38, 2009, doi: 10.1007/978-3-642-02830-4_4.
8. Y. Chen, B. Luo, Y. Chen, G. Liang, and X. Wu, “A Real-time Dynamic Hand Gesture Recognition System Using Kinect Sensor *,” pp. 2026–2030, 2015.
9. D. Ludl, T. Gulde, and C. Curio, “Simple yet efficient real-time pose-based action recognition,” in 2019 IEEE Intelligent Transportation Systems Conference, ITSC 2019, 2019. doi: 10.1109/ITSC.2019.8917128.
10. H. H. Pham, L. Khoudour, A. Crouzil, P. Zegers, and S. A. Velastin, “Exploiting deep residual networks for human action recognition from skeletal data,” Computer Vision and Image Understanding, 2018, doi: 10.1016/j.cviu.2018.03.003.
11. V. Silva, F. Soares, J. Sena Esteves, and G. Vercelli, “Human action recognition using an image-based temporal and spatial representation,” in 2020 12th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), 2021, pp. 1–6. doi: 10.1109/ICUMT51630.2020.9222408.
12. Y. Du, W. Wang, and L. Wang, “Hierarchical recurrent neural network for skeleton based action recognition,” in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2015. doi: 10.1109/CVPR.2015.7298714.
13. C. Li, Y. Hou, P. Wang, and W. Li, “Joint Distance Maps Based Action Recognition with Convolutional Neural Networks,” IEEE Signal Process Lett, vol. 24, no. 5, pp. 624–628, 2017, doi: 10.1109/LSP.2017.2678539.
14. S. Song, C. Lan, J. Xing, W. Zeng, and J. Liu, “An end-to-end spatio-temporal attention model for human action recognition from skeleton data,” in 31st AAAI Conference on Artificial Intelligence, AAAI 2017, 2017.
15. “M5StickC ESP32-PICO Mini IoT Development Kit | m5stack-store.” https://shop.m5stack.com/collections/m5-controllers/products/stick-c (accessed Feb. 27, 2023).
16. A. Calado, V. Silva, F. Soares, and P. Novais, “Ball Detection for Boccia Game Analysis,” in 2019 6th International Conference on Control, Decision and Information Technologies (CoDIT), 2019, pp. 1468–1473.
17. M. Lin, Q. Chen, and S. Yan, “Network in network,” in 2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings, 2014.
18. OpenCV, “OpenCV: Deep Neural Networks (dnn module),” 2020. https://docs.opencv.org/master/d2/d58/tutorial_table_of_content_dnn.html (accessed Apr. 08, 2020).