Indonesian Journal of Electrical Engineering and Computer Science
Vol. 28, No. 1, October 2022, pp. 98~105
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v28.i1.pp98-105

Implementation of a sensor node for monitoring and classification of physiological signals in an edge computing system

Ricardo Yauri[1], Antero Castro[2], Rafael Espino[2], Segundo Gamarra[3]
[1]Faculty of Systems and Informatic Engineering, Universidad Nacional Mayor de San Marcos, Lima, Perú
[2]Faculty of Engineering, Universidad Tecnológica del Perú, Lima, Perú
[3]Faculty of Electrical and Electronic Engineering, Universidad Nacional de Ingeniería, Lima, Perú

Article Info

Article history:

Received Mar 30, 2021
Revised Jul 20, 2022
Accepted Jul 29, 2022

Keywords:

Artificial intelligence
Edge computing
Electrocardiographic
Internet of things
Vector support machine
WebSocket

ABSTRACT

We describe the design and development of sensor nodes, based on Edge computing technologies, for the processing and classification of events detected in physiological signals such as the electrocardiographic signal (ECG is the electrical signal of the heart), temperature, heart rate, and human movement. The edge device uses a 32-bit Tensilica microcontroller-based module with the ability to transmit data wirelessly using Wi-Fi. In addition, algorithms for classification and detection of movement patterns were implemented to be implemented in devices with limited resources and not only in high-performance computers. The Internet of Things and its application in smart environments can help non-intrusive monitoring of daily activities by implementing support vector machine (SVM is a machine learning algorithm) for implementation in embedded systems with low hardware resources. This paper shows experimental results obtained during the acquisition, transmission, and processing of physiological signals in a edge computing system and their visualization in a web application.

Corresponding Author:

Ricardo Yauri
Faculty of Systems and Informatic Engineering, Universidad Nacional Mayor de San Marcos
Germán Amézaga 375, Lima, Perú
Email: ryaurir@unmsm.edu.pe

1. INTRODUCTION

Currently, devices that implement edge computing solutions usually using wireless transmission technologies (ZigBee, Bluetooth, Wi-Fi) [1], [2] and are applied in different areas using the benefits of the development of hardware and software solutions that enable data processing local and remote [3], [4]. In the specific case of the process of detecting events and movement patterns, it is usually carried out in computer systems with high hardware resources in cloud computing applications, so considerable energy consumption is required to transmit data to these services [5], [6]. In the case of physiological signal monitoring systems, these perform the recording and observation of biological signals but not in real time [7], [8]. When a sensor node detects events in its environment and performs intelligent actions analyzing this data, the concept of cognitive systems is involved in edge computing applications [9], [10] these devices being capable of understanding the inputs they receive through sensors, interpreting them in the context of the event and deducing some logical and rational response [11], [12].

The objective of this paper is to design and implement a machine learning algorithm for the classification of movement patterns using support and deployment vector machines in an embedded system (sensor node) with low hardware resources, generating design criteria, construction and deployment of edge computing solutions. A set of hardware and software technologies is proposed to monitor the health status of a person. The sensor node uses sensors to acquire electrocardiographic signals (ECG), temperature, heart rate and the detection of movement patterns, transmitting the information to a web application using internet of things (IoT) communication protocols based on WebSockets.

2. RESEARCH METHOD

2.1. Sensor node

The sensor node is developed to work with a battery and devices with low power consumption. For this case, the sleep mode technique is used to turn off the components of the sensor node that aren't used between data transmission periods [13]. The sensor node has an acquisition and processing stage. Then a wireless transmission stage and power control is used with a ESP8266 microcontroller [14], [15] (Figure 1).

Each sensor node has been designed to acquire data from a single sensor. The ECG signal is acquired using an AD8232 integrated circuit which integrates a signal coupling stage. The DS18B20 temperature sensor allows information to be acquired using a one-wire digital communication protocol. Besides, an operational amplifier is used to adapt the light signal that has been captured through a photodiode inside the pulse sensor [16]. Table 1 shows the monitoring frequencies of the physiological signals [17], [18]. The data is transmitted using Wi-Fi technology through Web communication protocols Figure 2. Stages are implemented to perform the sleep mode configuration processes and the deactivation of hardware components after performing the transmission.

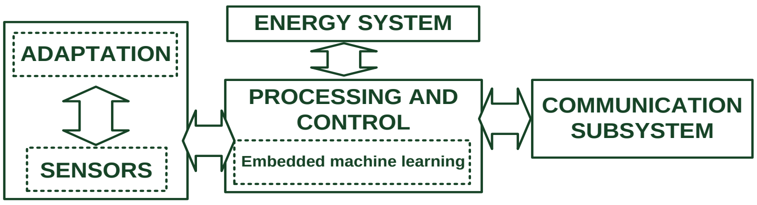


Figure 1. Edge computing system in a sensor node

Table 1. Physiological signals [18]

Parameter	Frequency range
ECG	0.5 Hz – 100Hz
Body temperature	1 HZ
Heartbeat	40 Hz
Electromyography	10Hz – 5KHz
Human Activity	0 Hz – 20Hz

Figure 2. Sensors nodes with wireless transmission

2.2. Data processing and detection of movement patterns in the sensor node

2.2.1. Generation of training data

For the implementation of the learning models, the GY801 inertial motion unit (IMU) is used as a generator of characteristics for the data set and the implementation of online classifiers [19]. For this, it is necessary to fulfill the following elements:

- Acquire inertial signals with a stable sample rate of up to 200 Hz.
- Label 3 kinds of movement: walking, running, and standing (rest).
- Use 1 triaxial sensor that contains an accelerometer.

- Use analog filtering built into the sensor device.
- Perform communication with a master device at a frequency higher than the sample rate.

2.2.2. Implementation of the SVM algorithm in the sensor node

The support vector machine (SVM) algorithm tries to divide data classes, with maximum separation, using a minimum number of data points [20], [21], using a hyperplane. Given a set of examples, the classes are labeled and then an SVM is trained to build a model that performs the prediction using new samples and support vectors obtained during model creation [22], [23]. The transformation of the data can generalize to another dimensional space, using a kernel equation, where the function "K" represents the operation between a data vector and the support vector, the result of which is multiplied with the output class value ($\propto_j y_j$) and applying the "sign" function to the result to obtain the classification result as described in [24], [25].

$$h(x_i) = \mathrm{sign}\left(\sum_{j=1}^{s}\left(\alpha_j y_j\, \mathrm{k}(x_j, x_i) + b\right)\right) \tag{1}$$

We consider three classes to classify which are: running, walking and static (standing) which we will call from now on as "co", "ca" and "is". Models based on a support vector machine Figure 3 were created, with the radial basis function kernel (RBF) [26], evaluating the one with the best behavior, adjusting the hyperplanes that will separate the classes.

```
In [15]:
from sklearn.svm import SVC
features, classmap = load_features('samp/')
```

```
In [16]:
X, y = features[:, :-1], features[:, -1]
classifier = SVC(kernel='rbf', gamma=0.001).fit(X, y)
#classifier = SVC(kernel='linear',gamma=0.001).fit(X, y)
```

Figure 3. SVM classification model created in python

To analyze the readings delivered by the sensors in real time, we arrange them in time windows with short periods. These data windows are stored in the random access memory (RAM) of the sensor node using 8-bit memory arrays for each element. Flash memory contains all the constants and parameters of the classifier model calculated offline, and which are necessary for real-time processing. Using the "micromlgen" library, the C++ language code for SVM models is generated (Figure 4). This resulting code will be copied to a header file ".h" that is imported from the main file. The model creation process is done using the python scikit-learn libraries in the Jupyter Notebook environment.

```
In [17]:
from micromlgen import port
c_code = port(classifier, classmap=classmap)
print(c_code)
```

```
#pragma once
namespace Eloquent {
    namespace ML {
        namespace Port {
            class SVM {
                public:
                    SVM() {
                    }

                    /**
                    * Predict class for features vect
                    */
                    int predict(float *x) {
                        float kernels[85] = { 0 };
                        float decisions[3] = { 0 };
                        int votes[3] = { 0 };
                        kernels[0] = compute_kernel(x
, -14.0  , -13.0  , 3.0  , -14.0  , -12.0  , 3.0  , -
4.0  , -16.0  , -10.0  , 4.0  , -16.0  , -10.0  , 4.0
```

Figure 4. Generation of the classification model code to program in the sensor node

A file called "model_rbf.h" is generated, which contains all the code sentences of the algorithm, using a feature vector as input. The characteristics extraction, processing and normalization process is the same as that used when the database was generated, only in this case, the "online classification" process, is carried out using the "classify" function show in Figure 5, which involves the process of:

- Prediction. Using the "predict" method.
- Conversion of the prediction to the class name, using the "classIdTxToName" method.
- Transmission of the class name to the WebSocket server. Using the "sendTXT" method.

The sensor node performs the online classification on the device, obtains the result which is sent to a Web application implemented with the Node-red software and node.js, using the WebSocket protocol. The obtained values are stored to later be compared with the real class.

```
void classify() {
    Serial.print("Detected gesture: ");
    String clas = clf.classIdxToName(clf.predict(features));
    Serial.println(clas);
    webSocket.sendTXT(clas + "\n");
}
```

Figure 5. Function for classification

3. RESULTS AND DISCUSSION

3.1. Acquisition and transmission data

The processes related to the data flow begin with the capture of data in the sensor node, the processing and detection of movement patterns and then the transmission of data to an application developed in Node-RED (programming by flows) that implements a service with the WebSocket technology. Figure 6 shows the communication scheme of the sensor node with the node-red application and node.js for classification verification.

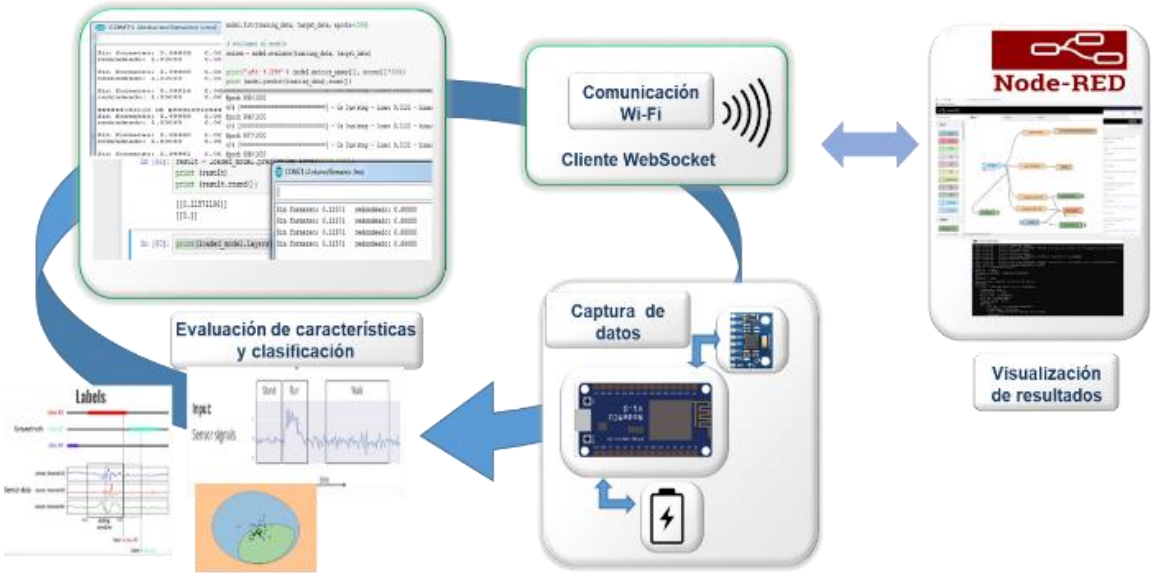


Figure 6. Communication scheme of the sensor node with node-red and node,js for classification verification

The sensor node firmware creates a WebSocket client to transmit data to the Node-RED web service Figure 7. The data is properly received by the web service and the web browser's developer console is used to verify it. During the data transmission stage, the average current consumption is recorded to be highest for heart rate data transmission, followed by the ECG signal and body temperature Figure 8.

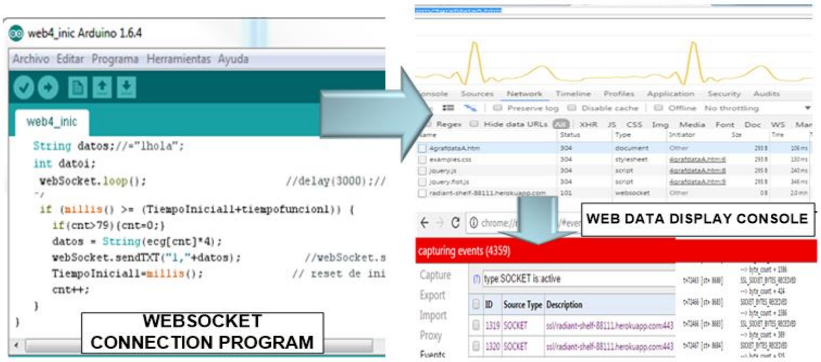


Figure 7. Data transmitted to the web service

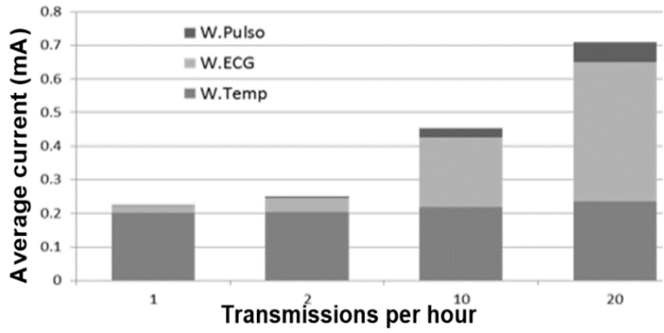


Figure 8. Current consumption by sensor nodes

3.2. Evaluation of the generated classification model

The generated model is implemented in the sensor node, comparing the score obtained in the classification with the score obtained from the model in the "training" and "testing" stage. Models with scaled data are considered due to the magnitudes generated by the Accelerometer have close values and do not affect the behavior of the model Figure 9.

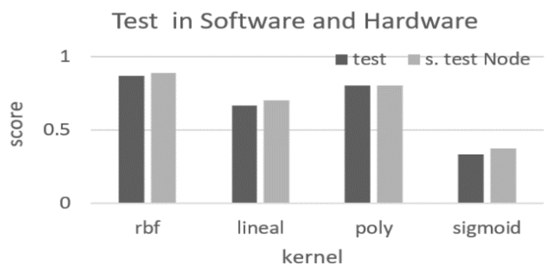
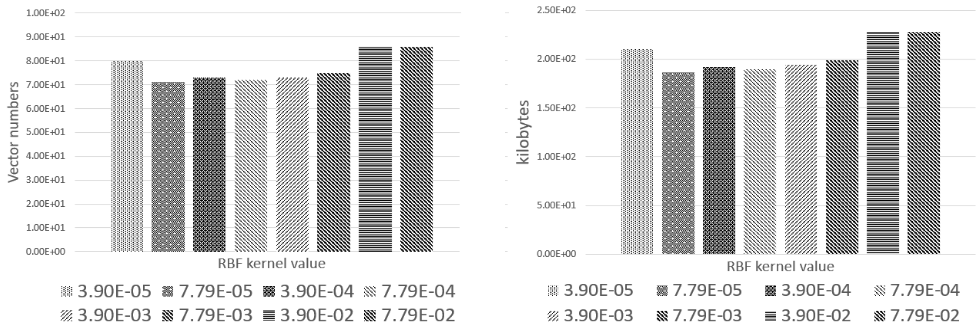


Figure 9. Comparison of the score obtained for different

The Figure 10(a) shows the number of support vectors in each model created with different types of radial basis function kernel (RBF). In this model, the "gamma" parameter is modified to observe its behavior during the "test" stage and select the most appropriate one according to the size of the generated file (Figure 10(b)). The model was evaluated with the kernel and value of "gamma" Figure 11 that had a better behavior to observe the score obtained during the classification in the sensor node and in the computer equipment where it was created. Figure 12 shows the relationship between the data referring to the memory size and the score value obtained by each of the learning algorithms based on the RBF kernel.



(a) (b)

Figure 10. Number of support vectors: (a) and generated file size and (b) for the RBF kernel-based learning model

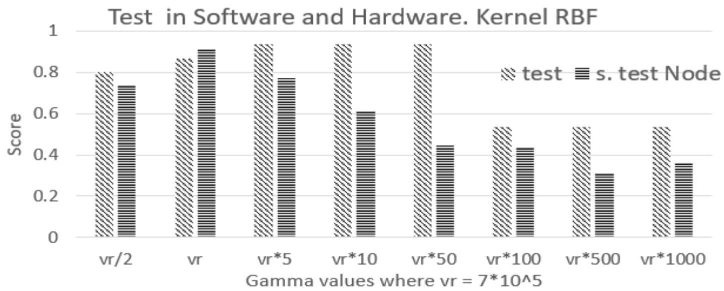


Figure 11. Comparison of the score of models evaluated in the training software and in the embedded system (Sensor node)

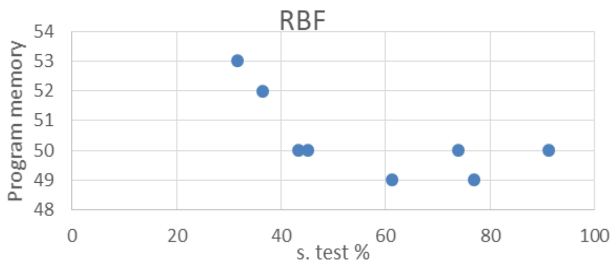


Figure 12. Comparison of program memory size with the RBF model score

4. CONCLUSION

The communication and current consumption test that Wi-Fi wireless technology is efficient depending on the number of bytes sent and the transmission frequency (the type of physiological sensor used affects the transmission frequency of data). Its observed that the highest current consumption of 0.7 mA occurs in the node with the pulse sensor due to the intensive transmission of data samples (80 bytes), which does not occur in the case of the temperature sensor, which transmits information of only 5 bytes, although the transmissions are increasing from 1 to 20 times per hour. This shows that the current consumption during data transmission remains the same value, until the number of bytes to be sent is increased to approximately 160 bytes.

In the case of the sensor node used for detection of movement patterns, several types of kernels were compared, where the RBF kernel has a score of 94%, compared to the Sigmoid kernel, polynomial and linear. These evaluated models were not made with scaled values because they consume more memory resources when implemented in an embedded system, although they have, as a strength, a better classification. The hyperparameter "Gamma", of the models based on the RBF kernel, directly affected the number of support vectors. Furthermore, is observed that the size of the code necessary to implement it can reach 260 Kbytes with

83 support vectors. Although some models have a greater number of support vectors, this causes an overfitting to the model in addition to a larger code size.

When evaluating the gamma parameter in the SVM model with RBF kernel, we obtain a better score for values in the range "vr*5" and "vr*50" (where vr = 7*10^5), which occurs when the model is created and configured on a computer equipment. However, when the model is displayed in the sensor node, obtain the best score occurs with a gamma value equal to "vr". The model can be trained and evaluated on a computer system with acceptable but not optimal results when implemented in a sensor node. In addition, the size of the memory space used as a variable for the implementation of the model has to be evaluated. Future works could carry out the evaluation of other wireless communication modules and their transmission characteristics with image or audio sensors. In addition, other classification algorithms, based on neural networks, can be implemented, evaluating the energy compromise between the energy used for the execution of the machine learning algorithms and the energy used for data transmission.

ACKNOWLEDGEMENTS

This work was developed as part of the Doctorate studies at the Faculty of Systems and Informatics Engineering of the Universidad Nacional Mayor de San Marcos and in the laboratories of INICTEL-UNI (National Institute of Research and Training in Telecommunications-Universidad Nacional de Ingeniería).

REFERENCES

[1] L. Betancur, "Redes de área corporal. Una perspectiva al futuro desde la investigación," *Sistemas y Telemática,* vol. 9, no. 16, pp. 11-30, 2011.
[2] P. K. D. Pramanik, S. Pal, and P. Choudhury, "Beyond automation: The cognitive IoT," *Artificial Intelligence Brings Sense to the Internet of Things*, Springer, pp. 1-37, 2018, doi: 10.1007/978-3-319-70688-71.
[3] R. Y. Rodriguez y M. R. Julcapoma, "Implementation of a sensor node for monitoring physiological signals with Websocket communication and data visualization in a node. js server over the Internet," in *2020 IEEE Engineering International Research Conference (EIRCON)*, 2020, doi: 10.1109/EIRCON51178.2020.9254049.
[4] A. Kumar, "Shiftry: RNN inference in 2KB of RAM," *Proceedings of the ACM on Programming Languages*, vol. 4, no. OOPSLA, pp. 1-30, 2020, doi: 10.1145/3428250.
[5] J. Mocnej, M. Miškuf, P. Papcun, and I. Zolotová, "Impact of edge computing paradigm on energy consumption in IoT," *IFAC-PapersOnLine*, vol. 5, no. 16, pp. 162-167, 2018, doi: 10.1016/j.ifacol.2018.07.147.
[6] F. Foukalas, "Cognitive IoT platform for fog computing industrial applications," *Computers & Electrical Engineering*, vol. 87, 2020, doi: 10.1016/j.compeleceng.2020.106770.
[7] D. M. R. Martín, "Contribución al análisis del movimiento humano aplicado a la identificación de posturas y bloqueos de la marcha en pacientes con Parkinson," Doctoral dissertation, Universitat Politècnica de Catalunya (UPC), Spain, 2014.
[8] M. Merenda, C. Porcaro, and D. Iero, "Edge machine learning for AI-enabled IoT devices: a review," *Sensors*, vol. 20, no. 9, p. 2533, 2020.
[9] B. Sliwa, N. Piatkowski, and C. Wietfeld, "Limitations, LIMITS: Lightweight Machine Learning for IoT Systems with Resource," in *2020 IEEE International Conference on Communications (ICC)*, 2020.
[10] R. Khusainov, D. Azzi, I. E. Achumba, and S. D. Bersch, "Real-time human ambulation, activity, and physiological monitoring: Taxonomy of issues, techniques, applications, challenges and limitations," *Sensors*, vol. 13, no. 10, pp. 12852-12902, 2013.
[11] A. V. Joshi, *Machine learning and artificial intelligence*, Springer, 2020.
[12] T. Goswami and U. B. Roy, "Predictive model for classification of power system faults using machine learning," in *TENCON 2019-2019 IEEE Region 10 Conference (TENCON)*, 2019, pp. 1881-1885, doi: 10.1109/TENCON.2019.8929264.
[13] D. Mengistu and F. Frisk, "Edge machine learning for energy efficiency of resource constrained IoT devices," in *SPWID 2019: The Fifth International Conference on Smart Portable, Wearable, Implantable and Disabilityoriented Devices and Systems*, 2019, pp. 9-14.
[14] R. Sanchez-Iborra and A. F. Skarmeta, "TinyML-enabled frugal smart objects: challenges and opportunities," *IEEE Circuits and Systems Magazine*, vol. 20, no. 3, pp. 4-18, 2020, doi: 10.1109/MCAS.2020.3005467.
[15] M. Schwartz, *Internet of Things with ESP8266*, Packt Publishing Ltd., 2016.
[16] K. Saleem, I. S. Bajwa, N. Sarwar, W. Anwar, and A. Ashraf, "IoT healthcare: design of smart and cost-effective sleep quality monitoring system," *Journal of Sensors*, 2020, doi: 10.1155/2020/8882378.
[17] M. J. Triana Laguna, "Reconocimiento de actividad personal empleando sensores a bordo de dispositivos inteligentes," Thesis, Universidad de Ibagué, Colombia, 2019.
[18] R. N. Meza, L. A. Vilca, and P. M. Vilcahuaman, "ECG con diagnostico por sistema experto y comunicación GPRS," *Universidad Nacional de san Agustín*, 2013.
[19] P. Valsalan, T. A. B. Baomar, and A. H. O. Baabood, "IoT based health monitoring system," *Journal of Critical Reviews*, vol. 7, no. 4, pp. 739-743, 2020.
[20] S. Raschka y V. Mirjalili, "Python machine learning: Machine learning and deep learning with Python, scikit-learn, and TensorFlow 2," *Packt Publishing Ltd*, 2019.
[21] A. Kowalczyk, "Support vector machines succinctly," *Syncfusion Inc*, 2017.

[22] B. S. Pattnaik, A. S. Pattanayak, S. K. Udgata, and A. K. Panda, "Advanced centralized and distributed SVM models over different IoT levels for edge layer intelligence and control," *Evolutionary Intelligence*, pp. 1-15, 2020.

[23] M. H. Abdul-Hadi, and J. Waleed, "Human speech and facial emotion recognition technique using SVM," in *2020 International Conference on Computer Science and Software Engineering (CSASE)*, 2020, pp. 191-196, doi: 10.1109/CSASE48920.2020.9142065.

[24] J. K. Nuamah, and Y. Seong, "Support vector machine (SVM) classification of cognitive tasks based on electroencephalography (EEG) engagement index," *Brain-Computer Interfaces*, vol. 5, no. 1, pp. 1-12, 2018, doi: 10.1080/2326263X.2017.1338012.

[25] A. V. Joshi, "Machine learning and artificial intelligence," *Springer Cham*, 2020, pp. 53-61.

[26] B. W. Z. Ma *et al.*,"Development of a support vector machine learning and smart phone Internet of Things-based architecture for real-time sleep apnea diagnosis," *BMC Medical Informatics and Decision Making*, vol. 20, no. 14, pp. 1-13, 2020, doi: 10.1186/s12911-020-01329-1.

BIOGRAPHIES OF AUTHORS

Ricardo Yauri is a Master of Science in Electronic Engineering with a Mention in Biomedical. He is associate professor at the National University of San Marcos (UNMSM) and PhD student in Systems Engineering. He has participated as a teacher in courses oriented to the Internet of things and applications in home automation and the Cisco academy for IoT. He was researcher at INICTEL-UNI in the Embedded Systems and Internet of Things research group. He developed research projects on the implementation of low consumption IoT devices that involve inference techniques, machine learning algorithms and computational intelligence. He can be contacted at ryaurir@unmsm.edu.pe.

Antero Castro is a graduate of the National University of Engineering of Peru (UNI) and Master in Information and Communication Engineering from the Beijing University of Aeronautics and Astronautics, China. With more than 10 years of professional experience and he has collaborated in the CHASQUI-I and Radioskaf-2 satellite projects. He currently works as a research professional at the National Institute of Training and Research in Telecommunications of the National University of Engineering (INICTEL-UNI). He is professor at the Universidad de Nacional de Ingeniería and Universidad Tecnológica del Perú. He develops research projects related to remote sensing and precision agriculture using multispectral and hyperspectral cameras. He can be contacted at c19828@utp.edu.pe.

Segundo Gamarra is Master in electrical engineering in the area of telecommunications and telematics from the State University of Campinas (UNICAMP)-Brazil. Undergraduate and Postgraduate Professor of the Faculty of Electrical and Electronics at the National University of Engineering. With more than 5 years of professional experience in electronic engineering solutions, he has also collaborated in the CHASQUI-I satellite project. Author of the paper Smart Campus IoT Guidance System for Visitors Based on Bayesian Filters and is currently a professor at the National University of Engineering. He can be contacted at sgamarra@uni.edu.pe.

Rafael Espino is Specialist in the development of electronic systems with solid knowledge in photovoltaic systems and extensive experience in implementing embedded firmware in electronic systems. He is a graduate of the Faculty of Electronics of the National University of Engineering He is currently a professor at the Universidad de Nacional de Ingeniería and Universidad Tecnológica del Perú. With aptitudes for teaching, research, and project management. He can be contacted at C22165@utp.edu.pe.