ISSN 2395-8618

Web Service for Automatic Detection of R-Peaks in Electrocardiograms

Cesar Fabian Reyes-Manzano, Miriam Laura Juarez-Hernandez, Juan Carlos Cisneros-Rasgado, Pablo Vera-González, Tania Jetzabel Contreras-Uribe

***Abstract*—The analysis of the electrocardiogram (ECG) is an essential tool for the diagnosis of cardiovascular diseases. Sometimes, it is necessary to carry out long-term recordings, even up to 24 hours, which must be carefully evaluated by specialists in order to generate an accurate diagnosis. In the case of extensive records, this task becomes monotonous, tedious and susceptible to errors, highlighting the need to implement Artificial Intelligence (AI) to automate this process, which would be a valuable support for health professionals. However, it is important to note that most devices used for ECG recording are low capacity, making them incompatible for running AI algorithms directly on them. An implementation of a web service for the identification of R-peaks in electrocardiograms is presented. This service performs the identification by means of a convolutional neural network, which after being trained reached a sensitivity of 0.99658 and a specificity of 0.99655. In this work, the main objective is to present a basic proposal to generate services that help to perform robust data analysis with high processing power consumption for low-resource devices such as microprocessors.**

***Index Terms*—R-peaks, CNN, serverless.**

I. Introduction

R-peak detection is a crucial task in the field of electrocardiogram (ECG) analysis, which involves identifying the prominent peaks in the ECG signal that correspond to the depolarization of the ventricles of the heart. Accurate detection of R-peaks is essential for various clinical applications, including arrhythmia diagnosis, heart rate variability analysis, and cardiac disease monitoring.

Traditionally, R-peak detection algorithms were based on heuristics and signal processing techniques, such as thresholding, template matching, and wavelet transforms. While these methods have shown reasonable performance, they often require careful tuning of parameters and may struggle with complex ECG signals that exhibit significant noise, baseline wander, and other artifacts [1-4].

The emergence of deep learning has revolutionized the field of biomedical signal processing, including ECG analysis.

Deep learning models, in particular neural networks, both convolutional (CNN) and recurrent (RNN), have demonstrated remarkable capabilities in machine learning of discriminative features from raw data, making them ideal candidates for R-peak detection [5, 6].

By training deep learning models on large annotated ECG datasets, it is possible to leverage the power of neural networks to automatically extract relevant features and identify R-peaks accurately. Deep learning-based R-peak detection algorithms have the potential to improve both the sensitivity and specificity of R-peak detection, leading to more reliable and efficient ECG analysis [7, 8].

Overall, the use of deep learning in R-peak detection presents an exciting avenue for advancing ECG analysis. By harnessing the power of neural networks and the abundance of annotated ECG data, deep learning models have the potential to revolutionize R-peak detection and contribute to more accurate and efficient cardiac monitoring systems [9].

Web services have changed the way online applications and systems interact with each other. Nowadays, in the digital era, where connectivity and collaboration are essential, web services play a key role in enabling different applications to communicate and share data over the Internet in an efficient and standardized way.

Furthermore, a web service is a technology that allows two or more applications, regardless of their location or programming language, to communicate with each other over the World Wide Web. These services follow specific principles and standards to ensure seamless and secure interoperability. A web service operates over HTTP (Hypertext Transfer Protocol) and uses data exchange formats such as XML (Extensible Markup Language) or JSON (JavaScript Object Notation) to transmit information in a structured manner.

These services are based on the REST (Representational State Transfer) architecture or, in some cases, on SOAP (Simple Object Access Protocol), which define how requests and responses are made [10].

The importance of web services lies in their ability to enable the integration of heterogeneous systems, meaning that applications developed in different programming languages or platforms can interact seamlessly. This has driven the creation of richer and more complex software ecosystems, from mobile applications and websites to large-scale enterprise systems [11].

In this context, this article has the purpose of presenting the way in which a CNN was implemented for the detection of R-peaks in electrocardiograms and how this algorithm can be executed through a web service on a web site.

This will be the basis for generating services that help to perform robust data analysis with high processing power consumption for low resource devices such as microprocessors.

Manuscript received on 13/04/2023, accepted for publication on 09/06/2023.

C.F. Reyes-Manzano, M. L. Juarez-Hernandez, J.C. Cisneros-Rasgado, P. Vera-González are with Tecnológico de Estudios Superiores de Ixtapaluca, División de Ingeniería en Sistemas Computacionales, Mexico (cesarrm5@hotmail.com).

T.J. Contreras-Uribe is with Tecnológico de Estudios Superiores de Ixtapaluca, División de Ingeniería Biomédica, Mexico.

ISSN 2395-8618

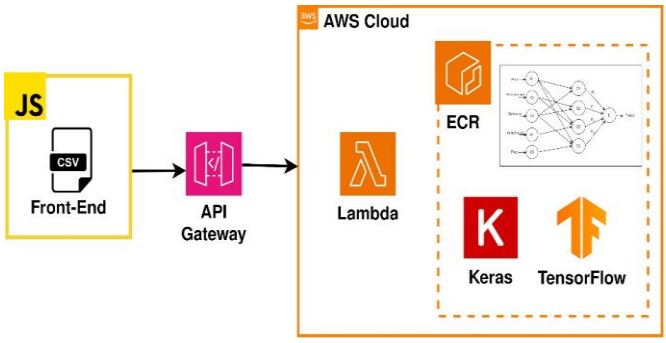

Fig. 1. System architecture

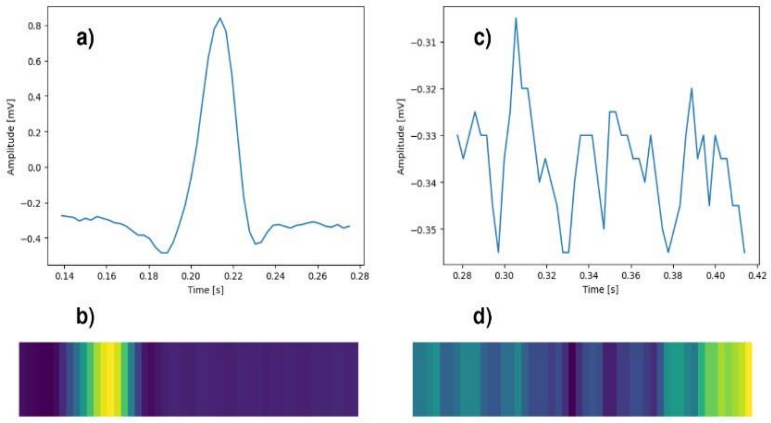

Fig. 2. Pre-processing of ECG signals. Figure 2a shows an example of a 139 ms window with a QRS complex and figure 2b shows the 50x10 pixel image generated by duplicating the window data 10 times. Figure 2c shows another window where a QRS complex is not found and its respective image in figure 2d

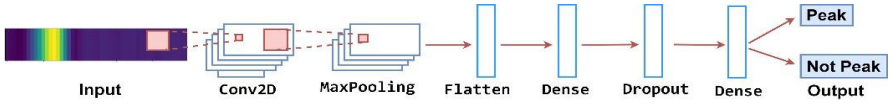

Fig. 3. CNN model

II. Methodology

A. Software Architecture

A serverless application was developed that allows the detection of R-peaks in ECG recordings. The application consists of a front-end and a back-end, as shown in Fig. 1. On the front-end, the web page was designed and programmed using HTML, CSS and JavaScript. This page connects to the back-end through an API programmed in the API Gateway service of Amazon Web Services (AWS).

The back-end uses a serverless architecture, which is made up of a Lambda application that executes the code of a previously trained CNN. All the necessary libraries for the execution (keras, tensorflow, numpy, json) are installed in an Amazon container, called Elastic Container Registry (ECR).

B. Convolutional Neural Networks (CNN) Training

The first stage of the proposal was the training of the convolutional neural network, for which it was important to select the database. Once the data was obtained, a preprocessing was performed, the network was trained, and the information was displayed. Each of the parts that make up this stage is described in detail below.

Dataset. The MIT-BIH arrhythmia database was used to train the CNN; it contains 48 recordings, each with a duration of half an hour, with the recording of two ECG channels of ambulatory patients (MLII, V1). For training, the MLII channel was used for recordings 100, 101 and 102, with a total of 6340 R-peaks in the recordings. The recordings were performed using a sampling rate of 360 samples/s per channel with a resolution of 11 bits in a range of 10 mV. The records were labeled by specialists for beat identification, these labels are stored in a .atr file within an array with the values of the time where a R-peak occurs. This database is focused on the analysis of arrhythmias, so the signals have been preprocessed to remove noise and no additional preprocessing is necessary [13].

Preprocessing. A Python script has been developed for the purpose of reading ECG records. From these records, channel V5 is extracted, and the information is organized into windows of 50 samples each, equivalent to 139 ms. With a sampling rate of 360 samples per second, this 139 ms. interval adequately captures the main shape of the QRS complex, which typically spans between 70 and 100 ms, as illustrated in Fig. 2a.

ISSN 2395-8618

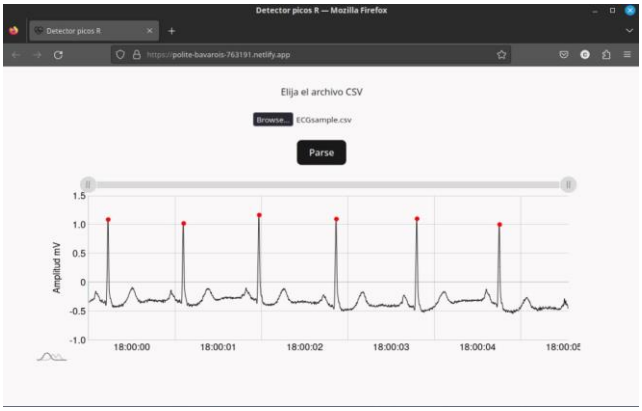

Fig. 4. Final website

TABLE 1
EVALUATION OF R-PEAK IDENTIFICATION ALGORITHM

Method	Sensitivity
Our work whit CNN	**0.9965**
Xiang et al. [18]	0.9977
Sarlija et al.[19]	0.9981
Chen et al. [20]	0.9981
Elgendi et al. [21]	0.9971
Lee et al.[22]	0.9969
Martinez et al.[4]	0.9913
Pan and Tompkins [3]	0.9913
Pandit et al. [2]	0.9965

Additionally, Fig. 2c displays a segment of the ECG signal without R peaks.

To create a 50 x10 image, the data window is replicated 10 times, as depicted in Fig. 2b and Fig. 2d, in order to generate redundant information and be able to implement a square kernel in the first layer of the CNN. This process is iterated throughout the entire record, resulting in the generation of 13,000 images for each record. Subsequently, each image will undergo evaluation by the CNN.

CNN Model. This Convolutional Neural Network (CNN) takes, as input, an image generated using a 50x10 sample window and produces a probability value indicating membership in either group one (peak R) or group zero (non-peak R). The implementation of the CNN utilized the TensorFlow library and features a 6-layer architecture, as shown in Fig. 3.

The first layer is a 2D convolution layer with 4 filters, a (5,5) kernel, an input size of (10,50), and a "relu" activation function. This layer conducts convolution in two dimensions on the image with the specified kernel for each filter, resulting in 4 output images. The second layer is a Max Pooling layer with a pool size of (2,2), reducing the size of the images by selecting the largest value from each pool group.

The third layer is a Flatten layer, transforming the input shape into a one-dimensional vector. The fourth layer is a Dense layer with a "relu" activation function and an output space dimensionality of 32. The fifth layer is a Dropout layer with a rate of 0.5, randomly setting input units to 0 with a frequency of the specified rate during training. Finally, the sixth layer is a Dense layer with a "softmax" activation function and an output space dimensionality of 2.

Training was performed in Colab with 12.7 GB of RAM and used a "BinaryCrossentropy" error function and a gradient descent optimizer function with learning rate parameters of 0.01 and impulse of 0.9. For training, the.atr annotation file was used to identify which windows had an R-peak (labeled with a value of one) and which did not (labeled with a value of 0). The hold-out 70-30 validation method was also used, i.e. 81,900 windows with 5,025 R-peaks for training and 35,100 windows with 1,902 Rpeaks for testing. For training, 75 epochs were proposed. The hold-out 70-30 validation method was repeated 10 times using different random seeds in each one.

Once CNN makes the prediction that an R-peak exists in the window, the maximum point of this window is searched to locate the exact point in time where the R-peak occurs.

Deployment. To perform the deployment of the CNN service, the trained network model was stored in a file. This file is read by a script located in the cloud, to obtain the CNN training parameters and make predictions. However, due to the weight of the Tensorflow library, a container image had to be implemented to associate the peak detection function with the Lambda function, which is a cloud service that allows executing code functions in serveless form [14].

For this purpose, an ECR container manager was used, where the container image was programmed using a docker-like file, where all the dependencies and code necessary for the classification of the peaks were placed, including the file with the trained model [15]. Once the image was implemented, it was linked to be called by the Lambda function, which in turn is triggered by the Gateway API service [16].

Once the REST API was implemented, the website was programmed using HTML, CSS and JavaScrip. This site has the function of reading the .csv file, decoding the ECG signal, separating it into windows of 50 samples, sending each window to the API to obtain its prediction and finally plotting the signal and the R-peaks found.

III. RESULTS

CNN was evaluated by counting how many times it correctly detected R peaks (true positives TP), how many times it did not detect R peaks where there were (False Negatives FN) and how many times it detected peaks where there were not (False Positives FP). These measurements were used to evaluate sensitivity (Se) and specificity (Sp) [17], which are defined as:

$$Sp = \frac{TN}{TN+FP}, \quad (1)$$

$$Se = \frac{TP}{TP+FN}. \quad (2)$$

The sensitivity and specificity obtained in the 10 evaluation cycles were averaged. The results yielded an average sensitivity of 0.99658 and a specificity of 0.99655, providing a comprehensive assessment of the model's performance.

The model, once trained, was exported to a POST API function, which is a RESTlike web service that can be accessed through the endpoint. In the body of this web request, the input parameters are included and in this case the data of the window to be evaluated in JSON format, as shown below:

ISSN 2395-8618

"signal": [-0.275, -0.28, -0.285, -0.305 -0.29, ...,-0.3,-0.28,-0.29].

This information is processed, and a response will be received in JSON format with two pieces of information: the first is the probability of belonging to the group containing the windows with R peaks, and the second is the probability of belonging to group two, which is the rest of the signal:

"result": ["0.9812707", "0.018729316"].

Finally, this API is consumed by a website that functions as a user interface. This website loads the CSV file and separates it into windows to be evaluated. After evaluating all the windows, it displays the results in a graph. An image of the website is shown in Fig. 4.

IV. Discussion

The values of some algorithms trained with the MIT-BIH database are shown in Table 4, where you can see the sensitivity achieved by them. Although our proposal is not the highest, we obtained results that competed with the others, showing differences in the range of thousandths.

The use of CNN has the advantage of not needing a pre-processing to obtain parameters, which helps to have a higher prediction speed once the network is trained. However, a disadvantage is that the parameters are unknown for classification, this can be overcome by delivering a data set robust enough to cover most of the possible cases. For example, for this case, include a window as R-peaks at different locations, at the center and at the edges, and in this way, you can control which cases you want to approve.

The system was implemented with a website and the architecture was designed to be used as a service, for example, in low processing power devices with Internet connectivity (IoT), because they only perform the REST request, and the processing is performed in the cloud. In addition, because the cloud system processes each window independently, with the help of JavaScript promises, the processing can resemble parallel processing, making it faster with costs like sequential processing.

While there is a cost associated with using AWS services, it is considered a better option than purchasing computer systems when services are used on a small scale. With serverless architecture, you only pay for the time it takes to process the request. This application is designed to analyze signals with a sampling frequency of 360 Hz.

In order to make this parameter variable, it is necessary to place it as an input parameter and it would imply having to make an adjustment in the size of the analysis window, which in this case is 50 samples, equivalent to 1.5 times the average duration of the QRS complex. The system demonstrates that it is feasible to perform ECG signal analysis on low capacity devices and even perform parallel requests to achieve greater speed. However, there are several algorithms for the detection of R peaks that can be implemented in these devices.

For this reason, it is proposed to use the same architecture for more complex analyses, such as arrhythmia diagnosis, heart rate variability analysis and cardiac disease monitoring, with the aim of justifying the suitability of the solution given the complexity of the cases.

V. Conclusion

This work proposes the design and implementation of a CNN for the detection of Rpeak with the MIT-BIH database, obtaining a sensitivity of 0.99658 and a specificity of 0.99655. These results show that the algorithm is competitive with resent algorithms. In addition, a web service for the detection of R-peaks in electrocardiogram signals was implemented in a web site using. The objective is that the service can be used by any device with an internet connection, mainly those with low processing power.

The system demonstrates that it is feasible to perform ECG signal analysis on low capacity devices and even perform parallel requests to achieve greater speed. However, there are several algorithms for the detection of R peaks that can be implemented in these devices. For this reason, it is proposed to use the same architecture for more complex analyses, such as arrhythmia analysis, with the aim of justifying the suitability of the solution given the complexity of the cases.

References

[1] J. Laitala, M. Jiang, E. Syrjala, E.K. Naeini, A. Airola, A.M. Rahmani, N.D. Dutt, and P. Liljeberg, "Robust ECG R-peak detection using LSTM," in *Proceedings of the 35th annual ACM symposium on applied computing*, pp. 1104–1111, 2020. DOI: 10.1145/3341 105.3373945.

[2] D. Pandit, L. Zhang, C. Liu, S. Chattopadhyay, N. Aslam, and C.P. Lim, "A lightweight QRS detector for single lead ECG signals using a max-min difference algorithm," *Computer Methods and Programs in Biomedicine*, vol. 144, pp. 61–75, 2017. DOI: 10.1016/j.cmpb. 2017.02.028.

[3] J. Pan and W.J. Tompkins, "A real-time QRS detection algorithm," *IEEE transactions on biomedical engineering*, vol. BME-32, no. 3, pp. 230–236, 1985. DOI: 10.1109/TBME.19 85.325532.

[4] J.P. Martínez, R. Almeida, S. Olmos, A.P. Rocha, and P. Laguna, "A wavelet-based ECG delineator: evaluation on standard databases," *IEEE Transactions on biomedical engineering*, vol. 51, no. 4, pp. 570–581, 2004. DOI: 10.1109/TBME.2003. 821031.

[5] P. Silva, E. Luz, E. Wanner, D. Menotti, and G. Moreira, "QRS detection in ECG signal with convolutional network," in R. Vera-Rodriguez, J. Fierrez, & A. Morales (eds), *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications. CIARP 2018. Lecture Notes in Computer Science, Springer, Cham*, vol. 11401, pp. 502–809, 2019. DOI: 10.1007/978-3-030-13469-3_93.

[6] M.M. Farag, "A self-contained STFT CNN for ECG classification and arrhythmia detection at the Edge," *IEEE Access* vol. 10, pp. 94469–94486, 2022. DOI: 10.1109/ACCESS.20 22.3204703.

[7] S. Vijayarangan, R. Vignesh, B. Murugesan, S. Preejith, J. Joseph, and M. Sivaprakasam, "Rpnet: A deep learning approach for robust R-peak detection in noisy ECG," in *2020 42nd annual international conference of the IEEE engineering in medicine & biology society (EMBC)*, pp. 345–348, 2020. DOI: 10.1109/EMBC44109.2020.9176084.

[8] V. Gupta, M. Mittal, and V. Mittal, "Performance evaluation of various pre-processing techniques for R-peak detection in ECG signal," *IETE Journal of Research*, vol. 68, no. 5, pp. 3267–3282, 2022. DOI: 10.1080/03772063.2020.1756473.

[9] N.A. Zermeño-Campos, D. Cuevas-González, J.P. García-Vázquez, R. López-Avitia, M.E. Bravo-Zanoguera, M.A. Reyna, and A. Díaz-Ramírez, "Péek: A cloud-based application

ISSN 2395-8618

for automatic electrocardiogram pre-diagnosis," *SoftwareX*, vol. 19, pp. 101124, 2022. DOI: 10.1016/j.softx.2022.101124.
[10] C. Ferris and J. Farrell, "What are web services?" *Communications of the ACM*, vol. 46, no. 6, pp. 31, 2003.
[11] F. Curbera, W. Nagy, and S. Weerawarana, "Web services: Why and how," in *Workshop on Object-Oriented Web Services-OOPSLA*, vol. 2001, 2001.
[12] E. Saavedra-Quijada, L.A. Medina-Muñoz, F. Morales-Solís, and G. López-Valencia, "Reconocimiento facial usando herramientas de IA de amazon web services y sistemas embebidos", *Research in Computing Science*, vol. 8, pp. 69–77, 2018.
[13] A.L. Goldberger, L.A. Amaral, L. Glass, J.M. Hausdorff, P.C. Ivanov, R.G. Mark, J.E. Mietus, G.B. Moody, C.K. Peng, and H.E. Stanley, "Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals," *Circulation*, vol. 101, pp. e215–e220, 2000. DOI: 10.1161/01.CIR.101.23.e215.
[14] Amazon Web Services, "Aws lambda," 2022.
[15] Amazon Web Services, "Amazon elastic container registry," 2022.
[16] Amazon Web Services, "Amazon API gateway," 2022.
[17] X. Lu, M. Pan, and Y. Yu, "QRS detection based on improved adaptive threshold," *Journal of healthcare engineering*, 2018.
[18] Y. Xiang, Z. Lin, and J. Meng, "Automatic QRS complex detection using two-level convolutional neural network," *Biomedical engineering online*, vol. 17, no. 13, pp. 1–17, 2018. DOI: 10.1186/s12938-018-0441-4.
[19] M. Sarlija, F. Jurisíc, and S. Popovíc, "A convolutional neural network-based approach to QRS detection," in *Proceedings of the 10th international symposium on image and signal processing and analysis, IEEE*, pp. 121–125, 2017. DOI: 10.1109/ ISPA.2017.8073581.
[20] H. Chen and K. Maharatna, "An automatic R and T peak detection method based on the combination of hierarchical clustering and discrete wavelet transform," *IEEE Journal of Biomedical and Health Informatics*, vol. 24, no. 10, 2825–2832, 2020. DOI: 10.1109/JBHI. 2020.2973982.
[21] M. Elgendi, "Fast QRS detection with an optimized knowledge-based method: Evaluation on 11 standard ECG databases," *PloS*, 2013. DOI: 10.1371/journal.pone.0073557.
[22] J. Lee, K. Jeong, J. Yoon, and M. Lee, "A simple real-time qrs detection algorithm," in *Proceedings of 18th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, IEEE*, vol. 4, pp. 1396–1398, 1996. DOI: 10.1109/IEMBS.1996.647473.