Computer Methods and Programs in Biomedicine 246 (2024) 108053

Contents lists available at ScienceDirect

Computer Methods and Programs in Biomedicine

journal homepage: www.elsevier.com/locate/cmpb

ECGMiner: A flexible software for accurately digitizing ECG

Adolfo F. Santamónica [a], Rocío Carratalá-Sáez [b,*], Yolanda Larriba [a], Alberto Pérez-Castellanos [c], Cristina Rueda [a]

[a] Depto. de Estadística e Investigación Operativa de la Universidad de Valladolid, Paseo de Belén 7, Valladolid, 47011, Castilla y León, Spain
[b] Depto. Informática de la Universidad de Valladolid, Paseo de Belén 5, Valladolid, 47011, Castilla y León, Spain
[c] Servicio de Cardiología, Hospital Universitario Son Espases, Instituto de Investigación Sanitaria de Baleares (IdISBa), Carretera de Valldemossa, 79, Palma, Illes Balears, Palma, 07120, Illes Balears, Spain

ARTICLE INFO

Keywords:
ECG
Images
Digitization
Software
FMM model

ABSTRACT

Background and objective: The electrocardiogram (ECG) is the most important non-invasive method for elucidating information about heart and cardiovascular disease diagnosis. Typically, the ECG system manufacturing companies provide ECG images, but store the numerical data in a proprietary format that is not interpretable and is not therefore useful for automatic diagnosis. There have been many efforts to digitize paper-based ECGs. The main limitations of previous works in ECG digitization are that they require manual selection of the regions of interest, only partly provide signal digitization, and offer limited accuracy.

Methods: We have developed the ECGMiner, an open-source software to digitize ECG images. It is precise, fast, and simple to use. This software digitizes ECGs in four steps: 1) recognizing the image composition; 2) removing the gridline; 3) extracting the signals; 4) post-processing and storing the data.

Results: We have evaluated the ECGMiner digitization capabilities using the Pearson Correlation Coefficient (PCC) and the Root Mean Square Error (RMSE) measures, and we consider ECG from two large, public, and widely used databases, LUDB and PTB-XL. The actual and digitized values of signals in both databases have been compared. The software's ability to correctly identify the location of characteristic waves has also been validated. Specifically, the PCC values are between 0.971 and 0.995, and the RMSE values are between 0.011 and 0.031 mV.

Conclusions: The ECGMiner software presented in this paper is open access, easy to install, easy to use, and capable of precisely recovering the paper-based/digital ECG signal data, regardless of the input format and signal complexity. ECGMiner outperforms existing digitization algorithms in terms of PCC and RMSE values.

1. Introduction

The increase in computing power, together with the popularization of algorithms to analyze bioelectrical signals, encourages the computer science community to design and implement software that supports the medical community. Bioelectrical signals arise from the human body produced by the displacement of ions in solution, such as the Electrocardiogram (ECG), Electroencephalogram, or neuronal Action Potentials, among others. ECGs in particular are routinely used in the clinic to identify heart anomalies. A standard ECG is recorded at 12 leads, where a lead is a glimpse of the heart's electrical activity from a particular angle. Moreover, a typical heartbeat is decomposed into five fundamental waves, respectively labeled as P, Q, R, S, and T. The main features used in medical practice are related to the location, amplitudes, and the time elapsed between waves (PR, QT, QRS, and ST intervals). To extract these quantitative values, the numerical signal is needed; however, often, only the graphic information of the signals generated by an electrocardiograph is available. Typically, the ECG system manufacturing companies provide ECG images, but store the numerical data in a proprietary format that is not interpretable and is not therefore useful for automatic diagnosis.

Since the later years of the past century, much effort has been put into digitizing paper-based ECGs. Many available ECG signal digitizing procedures base their analysis on calculating the cardiac rhythm (or heartbeat interval) to tentatively detect arrhythmia. On this matter, we find such works as [1,2], which measure the distance between

---

* Corresponding author.
E-mail addresses: adolfo.fernandez.santamonica@estudiantes.uva.es (A.F. Santamónica), rocio@infor.uva.es (R. Carratalá-Sáez), yolanda.larriba@uva.es (Y. Larriba), alberto.perezcastellanos@ssib.es (A. Pérez-Castellanos), cristina.rueda@uva.es (C. Rueda).

https://doi.org/10.1016/j.cmpb.2024.108053
Received 12 April 2023; Received in revised form 13 December 2023; Accepted 27 January 2024

Available online 3 February 2024
0169-2607/© 2024 The Author(s). Published by Elsevier B.V. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

the different heart peaks, as well as their number, along the time interval being measured. Other works digitize ECG signals with specific purposes, such as [3], presents a Matlab-based toolkit to extract certain parameters widely used in cardiology [4], including the duration of the PR interval, the QT interval, the QRS complex, and the ST interval. Alternatively, some authors offer tools that present a wider focus, attempting to provide general digitized information of the ECG signal that can be analyzed later, such as [5]. Particularly, in that work, the authors present a Matlab software, that stores the ECG signal as a binary file describing an image, as well as the demographic information. However, it requires manual user intervention as also happens in [6]. Aligned with this work, in [7], the authors present a Matlab software, but this time it has only been validated on simulated ECGs, which prevents a fair comparison with other works. More recently, [8] developed an open-access, fully automated algorithm; however, it employs non-public databases for validation and the proportion of discarded images is high. Furthermore, the use of neural networks has also been explored in such works as [9], which focuses on ECG digitization and its conversion into 1-D signals, leveraging deep learning-based binarization and diagnosis algorithms. A full review regarding digitization algorithms have been recently published in [10].

The motivation of this work is to offer open-source software to precisely digitize ECGs. In the particular context of our research group, the ECGMiner will form an ensemble with the FMM (Frequency Modulated Möbius) approach for ECG automatic interpretation, recently developed in [11]. The combination of ECGMiner plus FMM allows ECG images to be digitized and analyzed in real-time, thus generating a very useful collection of parameters and features of great interest in the clinic.

The rest of the paper is structured as follows. In Section 2, we describe the ECGMiner software, detailing both its backend and frontend. In Section 3, we detail the validation process and the performance results. Finally, we end with the Discussion and Conclusions, in Sections 4 and 5, respectively.

2. Methods

The potential of the ECGMiner software is twofold: On the one hand, at the backend, the digitizing process is precise and fast; on the other, the frontend consists of a Graphical User Interface (GUI) that is both simple and user-friendly.

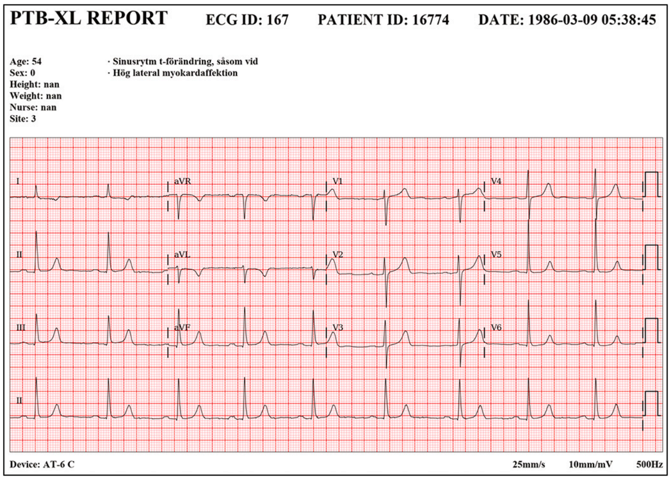
In this section, we first detail the format and components of the ECG images to digitize (Section 2.1); then we present the ECGMiner software frontend, describing its GUI (Section 2.2); following that, we explain the software backend, describing the collection of algorithms that have been implemented (Section 2.3); and finally, we provide some details on how to download the software (Section 2.4).

2.1. The input data to process: ECG images

Typically, an ECG image (see the example in Fig. 1) is composed of two differentiable regions: A grid with all the signals, and the frame that surrounds it, which contains all the metadata and demographic information of the patient to whom the ECG corresponds. The grid where the leads are plotted is usually printed in red or magenta, with the signals overlapped and printed in black or blue.

There exist different ways of representing the ECG leads. The most widely used format is the 12-lead ECG, which is usually displayed in 3×4 mode (3 rows and 4 columns), and chronologically located in the grid. This means that, for a typical 10-second long ECG, in 3×4 standard format, the first 2.5 seconds of the record correspond (from top to bottom) to the leads I, II, and III; from 2.5 to 5 seconds to the leads aVR, aVL, and aVF; from 5 to 7.5 seconds, to the leads V1, V2, and V3; and the last 2.5-second interval shows the leads V4, V5, and V6. This distribution has been schematically represented in Fig. 2a.

There exist variations of the 12-lead format. For example, the leads can be recorded distributed in 6×2 or 12×1 grids, or following specific formats such as the Cabrera format (see Fig. 2b).



PTB-XL REPORT   ECG ID: 167   PATIENT ID: 16774   DATE: 1986-03-09 05:38:45

Age: 54
Sex: 0
Height: nan
Weight: nan
Nurse: nan
Site: 3

· Sinusrytm t-förändring, såsom vid
· Hög lateral myokardaffektion

Device: AT-6 C   25mm/s   10mm/mV   500Hz

Fig. 1. Example of a printed ECG record from the PTB-XL database.

$$\begin{pmatrix} I & aVR & V1 & V4 \\ II & aVL & V2 & V5 \\ III & aVF & V3 & V6 \end{pmatrix} \qquad \begin{pmatrix} aVL & II & V1 & V4 \\ I & aVF & V2 & V5 \\ -aVR & III & V3 & V6 \end{pmatrix}$$

(a) Standard 12-lead format.   (b) Cabrera format.

Fig. 2. Common ECG display formats: Standard format on the left and Cabrera format on the right.

Moreover, some ECGs incorporate up to 3 rhythm leads, which are placed at the bottom of the grid and take up the total recording time (see the fourth row in Fig. 1). These signals are useful to locate wave features and measure, among others, heart rate from RR intervals. As they are an extended version of any of the leads, it is also useful to digitize them.

The small rectangular areas of the signals located at the end or at the beginning are called *reference pulses*. They do not belong to the signal itself as such, but they are used to mark the 0 mV (millivolts) and 1 mV reference values of each of the signals. See the rectangular peaks at the end of each row in Fig. 1.

Regarding the ECG frame, there is practically no consensus about the information it should include, nor about how to display it. Commonly, it contains some patient metadata, such as demographic information (name, age, gender, etc.) and clinical measures (heart rate, QRS complex, PR interval, etc.). Furthermore, the ECG frame can also be used to include the cardiologist's notes.

2.2. Frontend: the GUI

For the sake of usability, we have provided the ECGMiner software with a GUI (see Fig. 3) that enables easy interaction with the software.

On the left side of the GUI, in the gray region, there is the settings section, composed of:

- `Input` parameters: Here the user can select the input format of the ECGs to digitize. This includes:
  - `Main layout`: The leads distribution on the main layout. In particular, the software allows 3x4, 6x2 or 12x1 formats.
  - `1st Rhythm strip`: Name of the lead corresponding to the first rhythm strip. It can be set to `None`.
  - `2nd Rhythm strip`: Name of the lead corresponding to the second rhythm strip. It can be set to `None`.
  - `3rd Rhythm strip`: Name of the lead corresponding to the third rhythm strip. It can be set to `None`.
  - `RP on Left/RP on right`: Select whether the reference pulses are located on the left or right of the grid.

  - `Cabrera format`: Mark if the displaying mode of the leads is the Cabrera format, instead of the standard format.
- `Output` parameters: This is where the user indicates details regarding the results generated. It includes:
  - `Change path`: Set the path where the result files will be stored.
  - `Metadata OCR`: If selected, the software processes the ECGs' metadata and stores them in a text file at the end of the execution.
  - `Interpolate`: If selected, the software will interpolate the signals to the number of observations indicated by the user.

In the middle of the GUI, four elements are useful for obtaining information while the digitization process is taking place, and also when it has been completed. From top to bottom:

- `ECG selector` and `view-box`: Navigate throughout the loaded ECGs to see them.
- `Progress bar`: While digitizing the ECGs, this bar shows the percentage of already completed digitizations.
- `Console log`: Showcase the time and names of digitized ECGs, or error messages. In case of a failure, the software will proceed to digitize the remaining ECGs.

On the right-hand side of the GUI, there are four icons to interact with the software execution. From top to bottom:

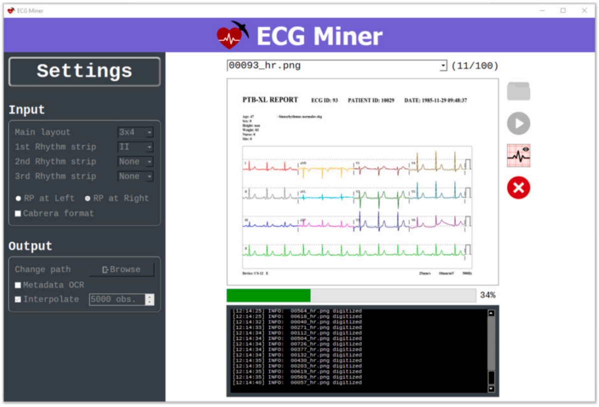
- `Load ECGs` (folder icon): By clicking here, the user specifies the path to the ECGs to digitize. Note that one or more ECGs can be selected.
- `Play`: This serves to start the digitizing process.
- `Switch view mode` (ECG signal with an eye icon): Switch between view modes for the selected ECG in the view box. In the digitized ECGs, the software showcases each lead detected by the extraction algorithm through colored lines, overlaying them on the actual leads. Furthermore, the view box displays each detected reference pulse with two dashed lines corresponding to 0 mV and 1 mV.
- `Cancel` (x icon): This can be used to stop and cancel the digitization process.

2.3. Backend: the digitization algorithm

At the backend, the ECGMiner software effectively conducts the ECG digitization. The process of digitizing an ECG image has been illustrated in Fig. 4. This flow can be globally described as follows:

1. `Image recognition`: Recognize the image composition. In this first step, the algorithm identifies which part of the image contains the metadata information, and which has the grid with the signals.
2. `Gridline removal`: Remove the gridline of the signal region, deleting anything that belongs to the background of the signal region. This step leaves only the lead signals with a clean background.
3. `Signal extraction`: The pixel coordinates that make up the signals are determined through an innovative ECG digitization technique. This approach involves maximizing connection plausibility based on a cost function defined simultaneously for all regions of interest. This not only enhances digitization accuracy but also considerably reduces computational time.
4. `Data storage`: Store the data in a CSV (Comma-Separated Values) file. Optionally, if requested by the user, it also exports the metadata information into a text file.

It has to be remarked that the ECGMiner takes advantage of parallel processing to speed up the calculations. As illustrated in Fig. 5, once the collection of ECG images to be digitized has been determined, they are split into batches. Then, multiple CPU threads are launched, each being in charge of digitizing one of the image batches.


ECG Miner
ECG Miner
Settings
Input
Output
Change path
Browse
PTB-XL REPORT ECG ID: 93 PATIENT ID: 10029 DATE: 1985-11-29 09:48:37
34%

00093_hr.png (11/100)

Main layout 3x4
1st Rhythm strip II
2nd Rhythm strip None
3rd Rhythm strip None
● RP at Left ● RP at Right
☐ Cabrera format
☐ Metadata OCR
☑ Interpolate 5000 obs.

```
[12:14:25] INFO:  00564_hr.png digitized
[12:14:25] INFO:  00618_hr.png digitized
[12:14:32] INFO:  00040_hr.png digitized
[12:14:33] INFO:  00271_hr.png digitized
[12:14:34] INFO:  00112_hr.png digitized
[12:14:34] INFO:  00504_hr.png digitized
[12:14:34] INFO:  00726_hr.png digitized
[12:14:34] INFO:  00377_hr.png digitized
[12:14:34] INFO:  00132_hr.png digitized
[12:14:35] INFO:  00430_hr.png digitized
[12:14:35] INFO:  00203_hr.png digitized
[12:14:35] INFO:  00619_hr.png digitized
[12:14:35] INFO:  00569_hr.png digitized
[12:14:40] INFO:  00057_hr.png digitized
```

Fig. 3. ECGMiner GUI.

2.3.1. Image recognition

At this first stage of the algorithm, the objective is to identify the location of the metadata information and the plotted signals in the ECG image. By intuition, one can assume that the signals will be placed inside the largest area of the image. Thus, this largest area is first identified as follows:

1. Simplify the image by extracting the edges of the global image to reduce the amount of data to be processed. This is done using the Canny Edge Detector operator [12], which uses a multi-stage algorithm to detect the edges of an image.
2. Obtain the bounded regions to construct polygons and discard the remaining loose edges. This is achieved by isolating every single contour by applying Suzuki's algorithm [13], which defines hierarchical relationships among the borders.
3. Transform the contours into rectangles using the Ramer–Douglas–Peucker algorithm [14,15], which reduces the number of points used in the approximation of a curve.
4. Take the rectangle with the largest area as the region that contains the grid with the signals.

2.3.2. Gridline removal

Grid lines hinder the digitization process and can be misleading. Moreover, the grid contains many points whose processing could considerably slow down the computation of the subsequent steps. Thus, once the grid region has been identified, the aim is to remove the grid, as well as any noise and small artifacts it may contain.

Based on the HSV (Hue, Saturation, Value) color model, an image can be seen as a multidimensional array; particularly, as an $M \times N \times 3$ array, with M rows, N columns, and 3 channels, respectively associated to *Hue*, *Saturation* and *Value*. The *Value* channel could be seen as the "color strength"; low values are associated with the signals (because they have dark colors), and the rest of the elements of the grid and other artifacts are represented by higher values. Considering 8-bit color graphics, each element contains a value between 0 and 255.

To detect the signal and remove the rest, the software converts the provided grid to a grayscale. It then applies a mask to the image pixels to remove or keep each of its pixels. This mask is determined by an automatic thresholding technique based on Otsu's method [16]. This method determines a value that serves to discriminate the pixels to be removed from those to be kept. In short, the algorithm returns a single intensity threshold that separates pixels into two classes, foreground and background. This threshold is calculated by maximizing inter-class variance on the intensity histogram (see Fig. 6). Once the threshold has been established, any pixel whose value is lower than or equal to it is set to 0, and any value higher than it is set to 1.

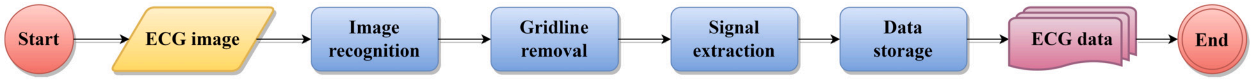


Fig. 4. Flowchart of the ECG digitization process.

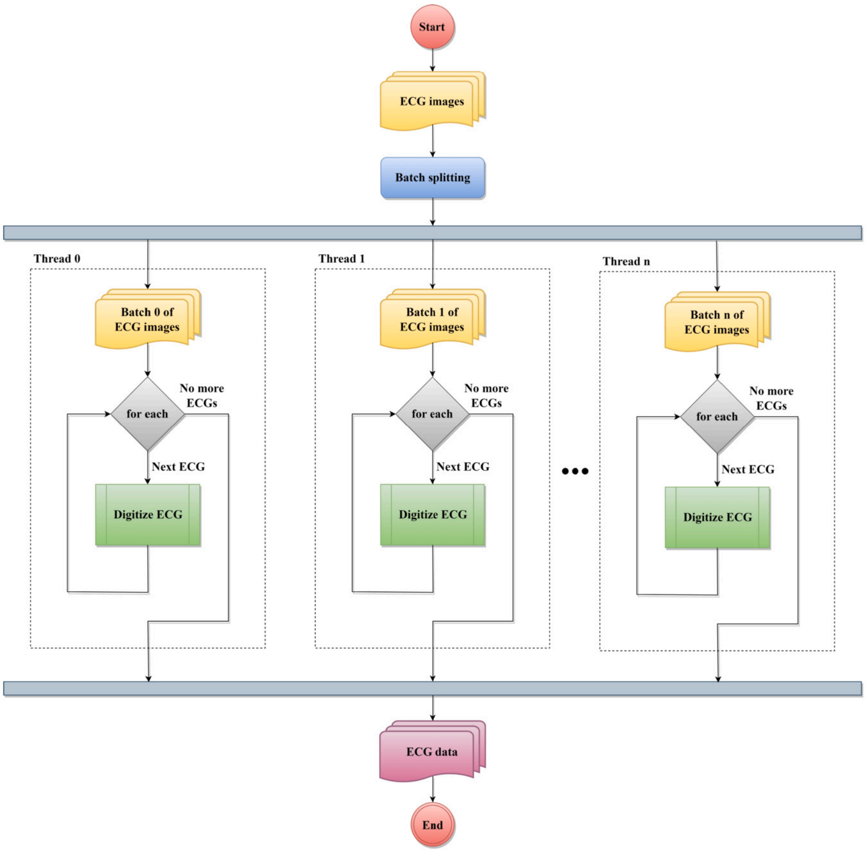


Fig. 5. Flowchart of ECG batch processing.

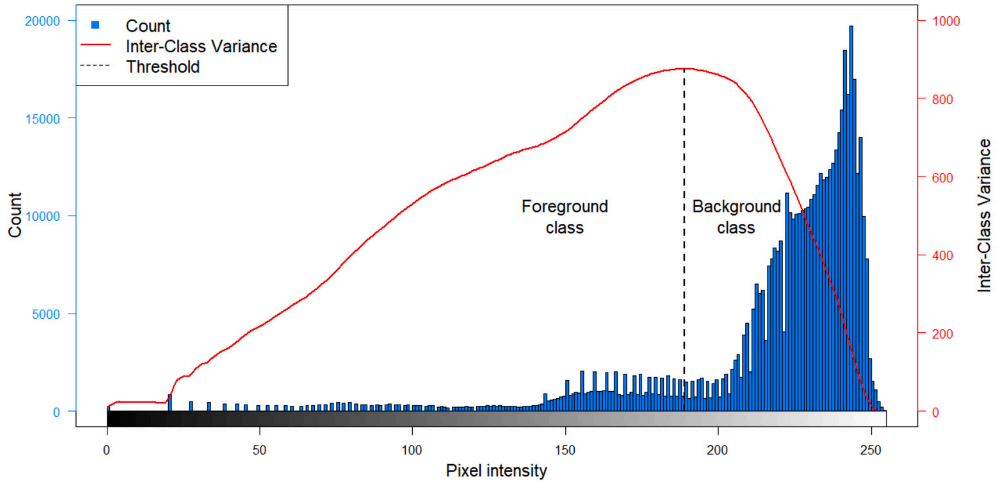


Fig. 6. A plot showing the results of applying Otsu's method to an ECG image.

2.3.3. Signal extraction

The signal extraction procedure (see Algorithm 1) consists of iterating, over the image pixel columns to store, the most likely connections between the different regions of consecutive black pixels (named clusters), based on a custom cost function, to identify the pixels belonging to each of the signals.

Algorithm 1 Signal extraction.

```
Input: $ecg$ (binary matrix) of dimensions $M \times N$
$LEN \leftarrow 2$
$SCORE \leftarrow 3$
$rois \leftarrow get\_rois(ecg)$
$roi\_n \leftarrow len(rois)$
$cache \leftarrow HashMap()$
for $col \in \{ini\_col \dots (N-1)\}$ do
    $clusters \leftarrow get\_clusters(ecg, col)$
    $prev\_clusters \leftarrow get\_clusters(ecg, col-1)$
    if $len(prev\_clus) = 0$ then
        continue
    end if
    for $c \in clusters$ do
        $cache[(col, c)] \leftarrow rep(\emptyset, roi\_n)$
        for $roi\_i \in \{0 \dots (roi\_n - 1)\}$ do
            $cost \leftarrow HashMap()$
            for $pc \in prev\_clusters$ do
                $node \leftarrow (col-1, pc)$
                $ctr \leftarrow ceil(mean(pc))$
                if $node \notin cache.keys()$ then
                    $val = (ctr, \emptyset, 1, 0)$
                    $cache[node] = rep(val, roi\_n)$
                end if
                $s \leftarrow cache[node][roi\_i][SCORE]$
                $d \leftarrow abs(ctr - rois[roi\_i])$
                $g \leftarrow gap(pc, c)$
                $cost[pc] \leftarrow s + d + M/5 \times g$
            end for
            $best \leftarrow cost.min()$
            $y \leftarrow ceil(mean(best))$
            $p \leftarrow (col-1, best)$
            $l \leftarrow cache[p][roi\_i][LEN] + 1$
            $s \leftarrow cost[best]$
            $cache[(col, c)][roi\_i] \leftarrow (y, p, l, s)$
        end for
    end for
end for
$waveforms \leftarrow backtracking(cache, rois)$
return waveforms
```

The first thing to do is to locate the regions of interest (ROI) in the image, which will be taken as the “centers” of each signal (see the magenta lines in Fig. 7b). Note that previous works use these ROIs to make a crop of each lead, either manually [6,7,17] or using an automatic lead cropping mechanism [18,19]. Manually cropping causes a huge bottleneck in the digitization process, as a lot of time has to be spent making the crops, so we discarded this option. The automatic cropping suffers from the risk of failing when digitizing ECGs that correspond to pathologies that impose certain conditions, such as the bundle branch block, making it almost impossible to correctly separate the signals into rectangles. Instead of making an independent crop for each lead, ECGMiner works with the entire image during the whole digitization process, allowing this type of ECGs to be digitized correctly (see Fig. 8).

Our software detects the ROIs by applying a sliding window with a size of 10 pixels over the whole image along the vertical axis and computing the standard deviation of each pixel. The centers of the windows with the highest deviation are identified as the ROIs, and they will be associated with the centers of each of the signals to be extracted (see the magenta crosses Fig. 7a). These peaks of standard deviation are calculated using the algorithm implemented via the SciPy library [20]. Next, to perform the connection process between the pixels, it iterates over each column of the image and the clusters of adjacent black pixels they contain. For each of them, a link is created with the cluster of the immediately preceding column that minimizes the cost function. This function considers the ROI (r) that is being taken as a reference, so the whole linkage process will be performed for each of the ROIs marked. For some ROI between a cluster (x_1) and another cluster from the previous column (x_2), the cost function is defined by Equation (1).

The first sum of the cost function is the *cumulative cost* (S) of the x_2 for ROI ***r***. D is the *Manhattan distance* from the center of x_2 to r, and G is the *gap* or vertical white space between x_1 and x_2. This gap will be 0 if they are in direct contact with each other. Note that this term is weighted by 20% of the number of rows M; which means that, although allowed, disconnections along the signal are severely penalized. This cost function is crucial for correctly identifying each lead when they overlap each other. Note that the algorithm does not include any particular consideration or specific procedure for overlapped leads, but always follows the described steps.

$$C(x_1, x_2, r) = S(x_2, r) + D(x_2, r) + \frac{M}{5} \times G(x_1, x_2) \tag{1}$$

To efficiently store and retrieve all these operations, a hashmap structure with `key-value` pairs is created to act as a cache. The `key` is each of the corresponding clusters and its associated `value` is a list of 4-length arrays (one array for each ROI). This list contains: 1) the row pixel coordinate of the midpoint of the cluster, 2) the cluster in the previous column with which it has been connected, 3) the total length of the signal up to that point, and 4) the cumulative cost of the previous cluster.

Once all the image pixel columns have been evaluated as described, the algorithm performs a backtracking traverse over the cache. This is done to obtain the longest paths with the minimal score for each ROI and store its coordinates as signal points.

Finally, the previously mentioned peak detection algorithm of Scipy [20] is applied over each signal to try to detect QRS complexes and outline their waves. This refinement helps to digitize the ECG waves with a little more accuracy, which is of considerable interest and clinical value.

2.3.4. Data storage

The first step of the data storage step consists of detecting the reference pulses from the signals, which are not desired to be stored. The coordinates of 0 mV and 1 mV of each pulse are memorized, and then these sub-sequences are deleted from the signals. The cleaned signals are divided into equal parts, each corresponding to a lead. This partition aligns with the user’s chosen input format selected at the beginning through the GUI, ensuring precise identification.

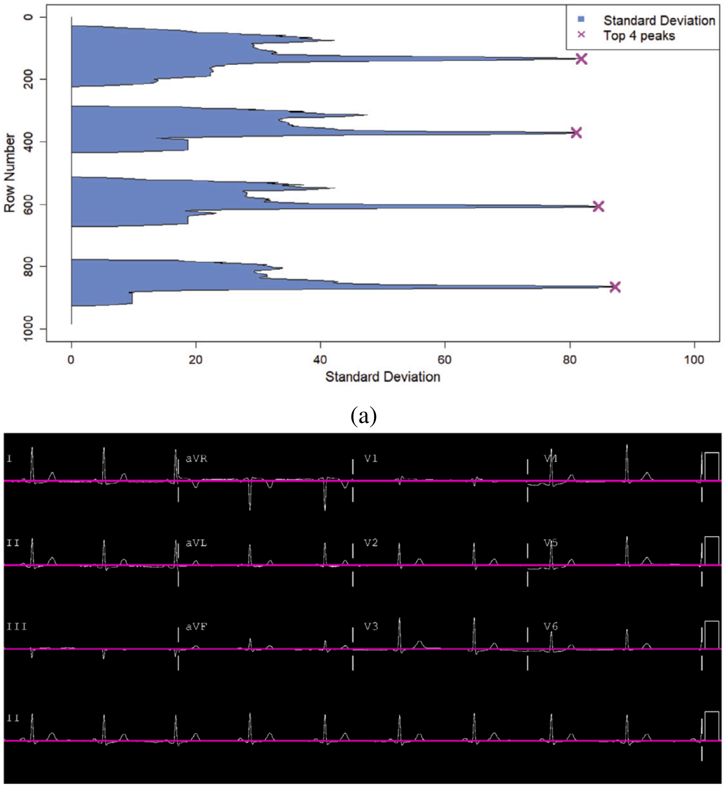
Before the signals are saved, a previous scaling is applied to each lead l, as shown in Equation (2), where $\mathbf{y}^{[l]}$ is the vector of the row pixel coordinates for lead l, $v_0^{[l]}$ and $v_1^{[l]}$ are, respectively, the row pixel coordinates of the 0 mV and 1 mV reference values for that lead, and $\boldsymbol{v}^{[l]}$ is a vector with the same length as $\mathbf{y}^{[l]}$ and all the elements equal to $v_0^{[l]}$.

$$\mathbf{s}^{[l]} = \frac{1}{v_0^{[l]} - v_1^{[l]}} \times (\boldsymbol{v}^{[l]} - \mathbf{y}^{[l]}). \tag{2}$$

In case the user chooses a specific number of observations, a linear interpolation is applied. This operation does not affect the shape of the signals if it is used as an upsampling technique for getting a higher number of observations than those originally represented.

Then, the ECGMiner stores the lead signals in a matrix-like structure and saves them in a CSV file, respecting temporal order and adding null values where no observations are available for a lead at a certain point in time.

Finally, if the user selected it in the ECGMiner GUI, an additional step is applied over the ECG to save the demographic information and



(a)

(b)

Fig. 7. (a) Example of the obtained row-based standard deviation for an ECG image, (b) ECG image with its ROIs highlighted. Magenta crosses and lines identify the centers.

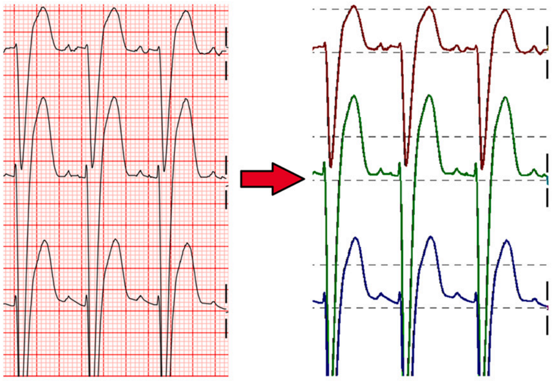


Fig. 8. Example of leads V1, V2, and V3 (vertically ordered) of a pathological ECG with bundle branch block before (left) and after (right) digitization. Note that the signals overlap.

the electrocardiogram metadata in a text file. For this, the software relies on Tesseract [21], an open-source optical character recognition engine that can output a unique string with the words structured in a paragraph, as a human would do when transcribing it.

Fig. 9 represents all the stages that are sequentially accomplished once the rectangle with the lead has been identified until the lead is digitized.

2.4. *Availability of ECGMiner*

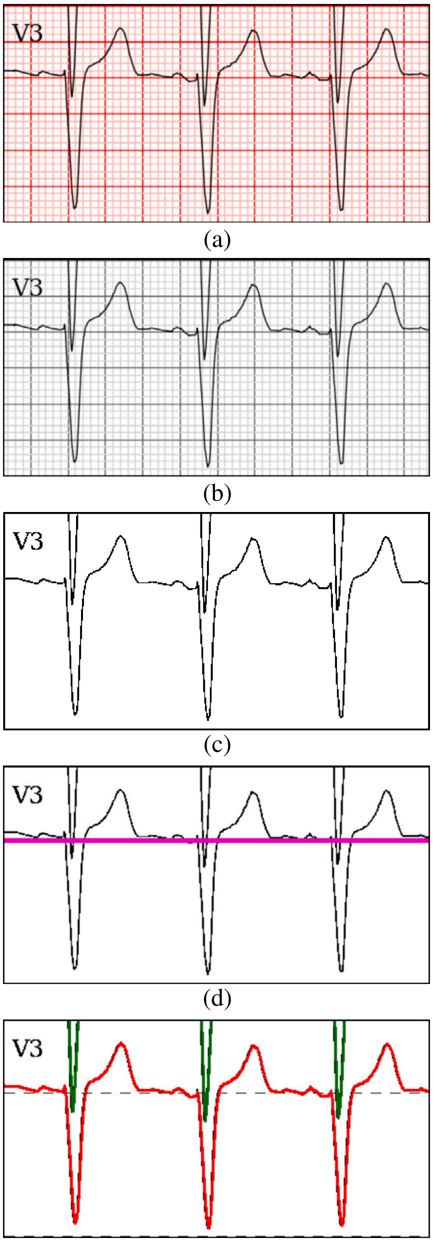
The ECGMiner software is fully open-source. Its source code, as well as the scripts used in the next section for validation, experiments, and the installation guide, is provided in the following GitHub repository: https://github.com/adofersan/ecg-miner.

3. Results

This section aims to validate ECGMiner using LUDB and PTB-XL, two databases of human ECGs publicly available at Physionet [22–24]. LUDB was recently built and includes 200 ECGs covering multiple diagnostics labels; while PTB-XL is a widely used and extensive database. We consider a subset of 2,203 out of the total of 21,837 ECGs in PTB-XL, stratified at the *Fold 10* set that is recommended for validation, as it contains revised ECGs labeled by experts [23]. Demographic and clinical characteristics of the subjects included in the study are similar across the databases, with approximately 40% healthy participants with normal ECGs (see Table 1). There exist notable differences between these databases. The LUDB data acquisition is from a unique device, while the PTB-XL source devices are diverse. Hence, the heterogeneity and complexity of the real ECG signals are better reflected in PTB-XL than in LUDB, including the presence of artifacts. To our knowledge, this is the largest database used in ECG digitization validation. Fig. 10 illustrates the differences between the ECGs of both databases. Subjects with a pacemaker were discarded, 4.5% and 1.32% from LUDB and PTB-XL, respectively.

For both databases, ECG records were acquired over 10 seconds across 12 leads and printed at 200 dpi resolution in standard format 3x4 display mode, with lead II as a rhythm strip following the American Heart Association standards [25]. Three validation tests were conducted:

- Test 1: Similarity between the actual and the digitized signals from LUDB (Section 3.1).
- Test 2: Similarity between the actual and the digitized P, R, and T marks from LUDB (Section 3.2).
- Test 3: Similarity between the actual and the digitized signals from PTB-XL (Section 3.3).



(a)

(b)

(c)

(d)

(e)

Fig. 9. Lead transformation sequence along the digitization: (a) Cropped ECG lead, (b) conversion to grayscale, (c) binarization (black/white), (d) ROI identified and highlighted in magenta, (e) digitized lead signal highlighted in red (part of another lead detected above highlighted in green).

Table 1
LUDB and PTB-XL database description: size, number of ECG used for validation, age (mean ± standard deviation), percentage of female individuals, and percentage of normal ECG diagnostic.

Attribute	LUDB	PTB-XL
Database size	200	2,203
Validation set size	191	1,857
Age	51.97 ± 19.25	60.92 ± 17.45
Female (%)	42.50	48.39
Normal ECG (%)	37.50	41.44

We use the Pearson correlation coefficient (PCC) and the root mean square error (RMSE) as validation result measures that are commonly used in digitization [5,6,18]. PCC measures the strength and direction of the linear relationship between the actual and digitized outcomes. It ranges from $[-1, 1]$, with 1 (resp. -1) indicating perfect positive (resp. negative) correlation, and 0 no correlation. RMSE is a positive value that quantifies the differences between the actual and digitized outcomes (in mV), where, the closer to 0, the better the digitized accuracy is.

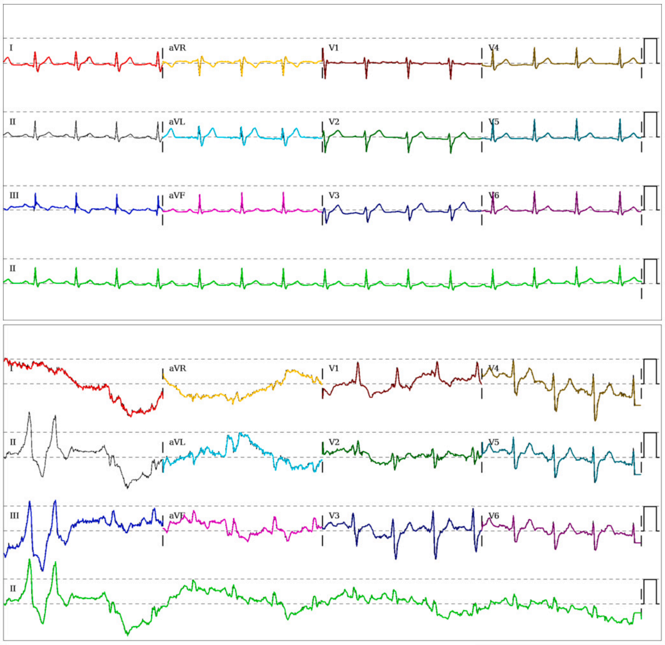


Fig. 10. Illustrative examples of 12-lead ECGMiner digitization taken from LUDB and PTB-XL. Top: Output for patient ID 185 in LUDB. Bottom: Output for patient ID 1157 in PTB-XL. The actual ECG signal is displayed in black, and different colored lines correspond to the digitized leads across signals.

For the different validation tests, ECGMiner was run on a computer equipped with an AMD Ryzen™ 5 3600 CPU @ 3.60 GHz (composed of 6 cores and 12 threads). ECGMiner was run on a Microsoft Windows 10 operating system using Python 3.10.7. Moreover, for architecture validation purposes, ECGMiner was executed on another computer featuring an Intel 11th Gen Intel(R) Core™ i5-1135G7 @ 2.40 GHz (comprising 4 cores and 8 threads), running on the Ubuntu 22 operating system with Python 3.10.7. We anticipate that the digitization accuracy rates remain comparable between the two computers.

Furthermore, we have also performed additional experiments with a dpi of 500. The findings revealed a marginal improvement in digitization results compared to 200 dpi, although the differences were nearly indistinguishable. At the end of this section, we provide a strong scaling analysis to demonstrate the parallel performance of our tool, where remarkable differences in terms of the elapsed time spent in the digitization are observed between 200 dpi and 500 dpi ECG images.

3.1. Test 1: similarity between the actual and the digitized signals from LUDB

An example of ECGMiner performance on LUDB is given in the top panel of Fig. 10. Table 2 shows the global PCC and RMSE values for the digitized ECGs from LUDB database. The PCC is consistently higher than 0.97 across all leads ($p < 0.001$) and the average RMSE is 0.010 mV ($p < 0.001$), indicating a substantial agreement between the actual and digitized signals. In particular, the agreement is almost perfect for the precordial leads (V1, V2, V3, V4, V5, and V6). These findings are sustained by the low values of RMSE. The results in terms of RMSE given for these leads in [6,8] are, on average, higher than twice and three times those obtained for LUDB, respectively.

Table 2
Test 1 (LUDB). Mean, standard deviation (SD), and p-value of PCC and RMSE across leads.

Lead	PCC			RMSE (mV)		
	Mean	SD	p-value	Mean	SD	p-value
I	0.979	0.013	< 0.001	0.024	0.005	< 0.001
II	0.988	0.009	< 0.001	0.017	0.005	< 0.001
III	0.971	0.022	< 0.001	0.021	0.007	< 0.001
aVR	0.987	0.008	< 0.001	0.019	0.004	< 0.001
aVL	0.977	0.024	< 0.001	0.019	0.006	< 0.001
aVF	0.981	0.016	< 0.001	0.017	0.005	< 0.001
V1	0.994	0.003	< 0.001	0.014	0.003	< 0.001
V2	0.993	0.007	< 0.001	0.013	0.003	< 0.001
V3	0.993	0.005	< 0.001	0.013	0.004	< 0.001
V4	0.992	0.004	< 0.001	0.015	0.003	< 0.001
V5	0.993	0.004	< 0.001	0.014	0.003	< 0.001
V6	0.994	0.004	< 0.001	0.011	0.003	< 0.001

3.2. Test 2: similarity between P, R, and T marks from LUDB

For this validation, all the actual and digitized versions of the ECGs from the LUDB validation set were renamed and randomly swapped. Next, the practitioner (A.P-C.) annotated the locations, in milliseconds (ms) of the marks of P, R, and T corresponding to the wave's peaks/troughs as vertical lines of different colors crossing the ECG paper. Finally, actual and digitized images were matched and the PCC and RMSE were computed to account for the differences between both annotation times. The results show a high performance in terms of PCC (with 0.999 average values) for P, R, and T; and RMSE (with 5.5 ms average values) for the annotations of the three marks.

3.3. Test 3: similarity between the actual and the digitized signals from PTB-XL

An example of ECGMiner performance on PTB-XL is given in the bottom panel of Fig. 10. The digitization task is much more difficult in this case because several signals are corrupted by artifacts. We identified two main drawbacks that obscure digitization. The first is the extreme overlapping with ECG paper borders or among leads by large voltage, especially marked in precordial leads. The second is the mixture of the alphanumeric lead's identification with the ECG signals (see examples in Fig. 11). Both issues were manually detected and discarded from the analysis. To be precise, 14.39% of the ECGs of PTB-XL were excluded for these reasons. Note that undergoing manual fine-tuning of the failed ECG images may correct most of these problems.

Average PCC and RMSE values for the digitized ECGs from PTB-XL database are shown in Table 3. Despite the intricate PTB-XL ECG signals, the PCC is consistently higher than 0.97 across all leads ($p < 0.001$). The average RMSE is 0.010 mV ($p < 0.001$), slightly larger than in the LUDB, a simpler and smallest database. For both databases results for lead I exhibit a slightly lower PCC. This can be attributed to the inherent noise typically associated with Lead I compared to other leads. While the RMSE values are slightly higher in leads V4 and V6 where overlapping is more likely. Despite these challenges, ECGMiner validation results, depicted in Table 4, consistently outperform, on average, those reported in [6,8], with notable superiority in terms of RMSE. Additionally, ECGMiner displays, compared to these alternatives, lower discarding rates and remarkably competitive digitization times which is key for its usage in the practice (see Table 4).

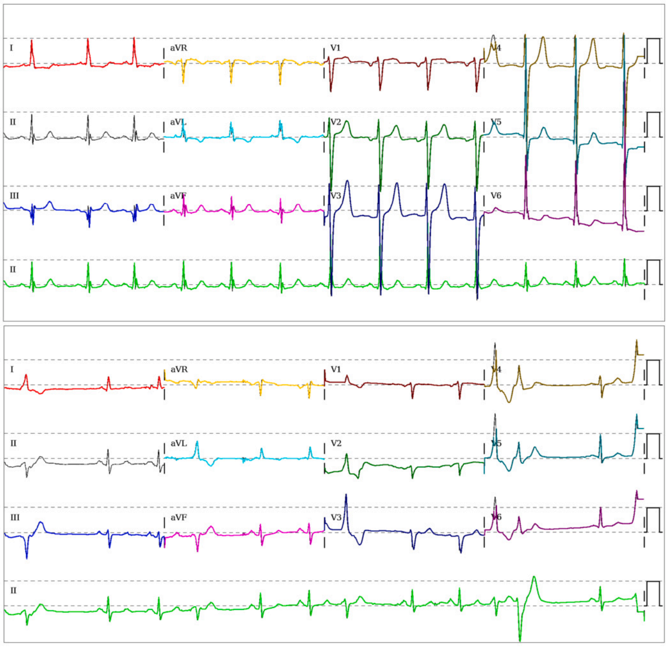


Fig. 11. Illustrative examples of ECGMiner fail in PTB-XL. Top: output for patient ID 3275 with the lead crossing in precordial leads. Bottom: output for patient ID 3805 with V1-V3 identification interposed with ECG signal. The actual ECG signal is displayed in black, and different colored lines correspond to the digitized leads across signals.

Table 3
Test 3 (PTB-XL). Mean, standard deviation (SD), and p-value of PCC and RMSE across leads.

Lead	PCC			RMSE (mV)		
	Mean	SD	p-value	Mean	SD	p-value
I	0.978	0.016	< 0.001	0.027	0.010	< 0.001
II	0.988	0.012	< 0.001	0.020	0.008	< 0.001
III	0.975	0.021	< 0.001	0.022	0.010	< 0.001
aVR	0.987	0.010	< 0.001	0.020	0.006	< 0.001
aVL	0.980	0.018	< 0.001	0.018	0.009	< 0.001
aVF	0.983	0.014	< 0.001	0.016	0.008	< 0.001
V1	0.995	0.003	< 0.001	0.017	0.006	< 0.001
V2	0.994	0.006	< 0.001	0.025	0.011	< 0.001
V3	0.993	0.006	< 0.001	0.028	0.013	< 0.001
V4	0.991	0.007	< 0.001	0.031	0.014	< 0.001
V5	0.992	0.007	< 0.001	0.026	0.012	< 0.001
V6	0.994	0.006	< 0.001	0.018	0.009	< 0.001

Table 4
Results comparison with alternative procedures. Database size, % of discarded ECGs, average ECG digitization time, PCC, and RMSE. To facilitate comparison, PCC and RMSE values are calculated across the leads I, II, and V1-V6 as outlined in [8].

	Test 1	Test 3	*Fortune et al.* [6]	*Wu et al.* [8]
Database size	200	2,203	80	923
Discarded ECGs (%)	4.50	14.39	50.00	44.20
ECG digitization time (s)	0.5-1	0.5-1	3–5	60-120
PCC	0.991	0.991	0.977	0.990
RMSE (mV)	0.015	0.024	0.030	0.040

3.4. Parallel performance

As we have previously stated, our software can digitize several ECG images in parallel. To evaluate the parallel performance of our tool, we have conducted a strong scaling analysis, measuring the scalability of our implementation when increasing the number of threads and maintaining a fixed amount of work. We have decided to conduct this analysis to evaluate the software scalability by employing the two databases already used in our validation tests: LUDB and PTB-XL (following the characterization shown in Table 1). For the performance analysis, we have employed a High Performance Computing (HPC) server that features two Intel(R) Xeon(R) Platinum 8160 CPU @ 2.10 GHz, with 24 Core Processors and 48 threads each. This system operates on CentOS 7

Table 5
Execution time (in seconds) of ECGMiner when using up to 96 threads for digitizing ECG images from LUDB and PTB-XL databases on an HPC server.

# threads	**LUDB**		**PTB-XL**	
	200 dpi	500 dpi	200 dpi	500 dpi
1	699.13	2391.41	7974.83	28348.60
2	359.32	1239.78	4076.25	14231.62
4	186.15	631.12	2078.06	7434.86
6	128.67	427.36	1417.51	5053.73
8	100.95	323.87	1084.69	3869.36
12	71.39	224.99	754.16	2661.31
16	58.17	177.48	596.10	1947.61
24	36.39	129.39	444.80	1371.70
32	27.69	107.79	300.92	1079.31
48	17.88	50.73	122.37	384.88
64	17.82	44.22	114.93	263.69
96	13.12	35.53	90.06	250.54

operating system, and ECGMiner has been run using Python 3.10.9. Table 5 reflects the execution time (in seconds) observed when digitizing LUDB and PTB-XL ECG images in an HPC server, using up to 96 threads.

The results show that the software follows an almost perfect efficiency when using up to 48 threads, and then it starts performing slightly worse. This is justified by the fact that, with more than 48 threads, hyperthreading is leveraged, causing a less efficient management of the resources.

4. Discussion

The ECGMiner software, introduced and validated in this paper, is open-access, user-friendly, and easy to install. It excels in accurately retrieving 12-lead ECG signal data, showcasing versatility across a spectrum of ECG formats.

4.1. Comparison with other works

The potential of ECGMiner is sustained by the versatility offered, the easy and accessible handling, and the automation of the whole digitization process. First, ECGMiner allows 12-lead digitization with single or multiple rhythm strips; as well as in Standard or Cabrera format, broadening the possibilities of similar works [3,17]. Second, the diverse outputs provided (CSV digitized data files, PNG files to compare raw and digitized images, and TXT files with metadata) enlarge its usability in medical practice. Third, ECGMiner software is open-free available. Fourth, we process ECGs without any user intervention to crop each lead, while this is manually done in other works [6,7]. Fifth, all the signals are extracted over the entire image, not over rectangular crops that may lose information from the signals, as done in [18,19]. Sixth, ECGMiner allows a collection of ECG images to be digitized, batch-processing them in parallel, unlike in [8]. This means it can fully leverage the computational system capabilities to offer digitized results faster. Finally, the software can be run on both Windows and Linux operating systems and the LUDB and PTB-XL databases are fully publicly available to facilitate reproducibility and comparisons.

ECGMiner feasibility is given in terms of PCC and RMSE, as having accurate, objective, and fully interpretable measures, in contrast to manual pixel distance-based differences or manual validations provided by other works, such as [6,7]. In addition, we considered two different databases, which include a variety of healthy, pathological, and artifacts ECGs. This is a key difference compared to other papers in the literature, where the validation set is homogeneous, with smaller sample sizes, and, in many instances, not publicly accessible [5,6,8,18]. This work proposes a validation of the digitization on the largest ECG set up to date. Finally, in terms of accurate measures, ECGMiner outperforms the results in [6,8].

4.2. Limitations

ECGMiner presents some limitations. First, the letters indicating the name of the leads could not be correctly removed using optical engines, as they were intermingled with the signals and could hardly be distinguished, and also because they had some Roman characters. On the other hand, an extreme overlap of the leads can lead the digitization process to fail. None of the alternatives to ECGMiner are capable of solving this issue, which we plan to deal with in the future.

4.3. Future research

The advantages derived from utilizing digitized data linked to ECG images extend beyond mere data representation. By leveraging digitized information, we gain the capability for automated diagnosis and in-depth research of various cardiac pathologies. In particular, drawing attention to details that might be overlooked by human observation. Furthermore, this approach creates opportunities for developing software to support medical diagnoses. The spectrum of the software extends from statistical methods to advanced machine learning algorithms and artificial intelligence (AI). This encompasses the training of deep neural networks and the fine-tuning of their parameters, ultimately enhancing their performance. A tailored version of this software can be integrated into clinical settings immediately after obtaining and digitizing the ECG image, resulting in nearly real-time diagnostic outcomes.

Specifically, we are integrating ECGMiner with the FMM_{ecg} approach to enable real-time digitization and analysis of ECG images. The FMM generates a comprehensive set of parameters and features that hold significant clinical relevance. Currently, this is being tested in a pilot study, utilizing images sourced from actual patients. A total of 175 images, representing consecutive patients seen at Dr. Pérez- Castellanos' cardiology service at Son Espases University Hospital from January 1, 2023, to June 30, 2023, have been digitized using ECGMiner. These images adhere to the standard 3x4 format commonly employed in the service. Notably, ECGMiner has successfully digitized 168 out of 175 images (96%), with only 7 cases (4%) exhibiting extreme overlapping. In addition, an advanced AI system, designed for the automatic interpretation of ECGs, is being developed as part of our ongoing initiatives.

5. Conclusions

ECGMiner is an open-access software for digitizing multiple paper-based ECGs simultaneously. Its minimalist user interface allows both cardiologists and researchers to use the tool with hardly any training. The validation results show very high correlations between the actual and the digitized signals. We can therefore conclude that the system is capable of very precisely recovering the data generated by the images provided by the usual electrocardiographs used in hospitals.

CRediT authorship contribution statement

Adolfo F. Santamónica: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. **Rocío Carratalá-Sáez:** Conceptualization, Formal analysis, Investigation, Methodology, Resources, Software, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing. **Yolanda Larriba:** Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Resources, Software, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing, Funding acquisition. **Alberto Pérez-Castellanos:** Conceptualization, Data curation, Formal analysis, Methodology, Resources, Validation, Writing – review & editing. **Cristina Rueda:** Conceptualization, Data curation, Formal analysis, Funding acquisition, Investigation, Methodology, Project administration, Resources, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing.

Declaration of competing interest

The authors declare that there are no known competing financial interests or personal relationships that could have appeared to influence this work.

Funding

The authors gratefully acknowledge the financial support received by the Spanish Ministry of Science, Innovation and Universities [PID2019-106363RB-I00 to C.R. and Y.L.].

References

[1] P. Reddy Gurunatha Swamy, S. Jayaraman, M. Girish Chandra, An improved method for digital time series signal generation from scanned ecg records, in: 2010 International Conference on Bioinformatics and Biomedical Technology, 2010, pp. 400–403.

[2] T.P. Exarchos, M.G. Tsipouras, D. Nanou, C. Bazios, Y. Antoniou, D.I. Fotiadis, A platform for wide scale integration and visual representation of medical intelligence in cardiology: the decision support framework, in: Computers in Cardiology, 2005, 2005, pp. 167–170.

[3] S. Mallawaarachchi, M. Prabhavi, N. Perera, Nuwan D. Nanayakkara, Toolkit for extracting electrocardiogram signals from scanned trace reports, in: 2014 IEEE Conference on Biomedical Engineering and Sciences (IECBES), 2014, pp. 868–873.

[4] K.E. Barrett, S.M. Barman, S. Boitano, H.L. Brooks, Origin of the Heartbeat & the Electrical Activity of the Heart, McGraw-Hill Education, New York, NY, 2018.

[5] L. Ravichandran, C. Harless, A.J. Shah, C.A. Wick, J.H. Mcclellan, S. Tridandapani, Novel tool for complete digitization of paper electrocardiography data, IEEE J. Transl. Eng. Health Med. 1 (2013) 1800107.

[6] J.D. Fortune, N.E. Coppa, K.T. Haq, H. Patel, L.G. Tereshchenko, Digitizing ecg image: a new method and open-source software code, Comput. Methods Programs Biomed. 221 (2022) 106890.

[7] V. Randazzo, E. Puleo, A. Paviglianiti, A. Vallan, E. Pasero, Development and validation of an algorithm for the digitization of ecg paper images, Sensors 22 (19) (2022).

[8] H. Wu, K. Haresh Kumar Patel, X. Li, B. Zhang, C. Galazis, N. Bajaj, A. Sau, X. Shi, L. Sun, Y. Tao, et al., A fully-automated paper ecg digitisation algorithm using deep learning, Sci. Rep. 12 (1) (2022) 20963.

[9] S. Mishra, G. Khatwani, R. Patil, D. Sapariya, V. Shah, D. Parmar, S. Dinesh, P. Daphal, N. Mehendale, Ecg paper record digitization and diagnosis using deep learning, J. Med. Biol. Eng. 41 (4) (2021).

[10] A. Lence, F. Extramiana, A. Fall, J.-E. Salem, J.-D. Zucker, E. Prifti, Automatic digitization of paper electrocardiograms–a systematic review, J. Electrocardiol. (2023).

[11] C. Rueda, A. Rodríguez-Collado, I. Fernández, C. Canedo, M.D. Ugarte, Y. Larriba, A unique cardiac electrophysiological 3D model. Toward interpretable AI diagnosis., iScience 15 (12) (2022).

[12] J. Canny, A computational approach to edge detection, IEEE Trans. Pattern Anal. Mach. Intell. PAMI-8 (1986) 679–698.

[13] S. Suzuki, K. Abe, Topological structural analysis of digitized binary images by border following, Comput. Vis. Graph. Image Process. 30 (1985) 32–46.

[14] U. Ramer, An iterative procedure for the polygonal approximation of plane curves, Comput. Graph. Image Process. 1 (1972) 244–256.

[15] D.H. Douglas, T.K. Peucker, Algorithms for the reduction of the number of points required to represent a digitized line or its caricature, Cartographica 10 (2) (1973) 112–122.

[16] N. Otsu, A threshold selection method from gray-level histograms, IEEE Trans. Syst. Man Cybern. 9 (1979) 62–66.

[17] F. Badilini, T. Erdem, W. Zareba, A.J. Moss, Ecgscan: a method for conversion of paper electrocardiographic printouts to digital electrocardiographic files, J. Electrocardiol. 38 (4) (2005) 310–318.

[18] M. Baydoun, L. Safatly, O.K. Abou Hassan, H. Ghaziri, A. El Hajj, H. Isma'eel, High precision digitization of paper-based ecg records: a step toward machine learning, IEEE J. Transl. Eng. Health Med. 7 (2019) 1–8.

[19] X. Sun, Q. Li, K. Wang, R. He, H. Zhang, A novel method for ecg paper records digitization, in: Computers in Cardiology, 2019, vol. 46, 2019.

[20] P. Virtanen, R. Gommers, T.E. Oliphant, et al., Scipy 1.0: fundamental algorithms for scientific computing in python, Nat. Methods 17 (2020) 261–272.

[21] R. Smith, An overview of the tesseract ocr engine, in: Ninth International Conference on Document Analysis and Recognition (ICDAR 2007), 2007, pp. 629–633.

[22] A. Kalyakulina, I. Yusipov, V. Moskalenko, A. Nikolskiy, K. Kosonogov, N. Zolotykh, M.I. Ludb, A new open-access validation tool for electrocardiogram delineation algorithms, IEEE Access 8 (2020) 186181–186190.

[23] P. Wagner, N. Strodthoff, R.-D. Bousseljot, F.I. Kreiseler, D. abd Lunze, W. Samek, T. Schaeffter, Ptb-xl, a large publicly available electrocardiography dataset, Sci. Data 7 (2020).

[24] A.L. Goldberger, L.A.N. Amaral, L. Glass, J.M. Hausdorff, P.Ch. Ivanov, R.G. Mark, J.E. Mietus, G.B. Moody, C.-K. Peng, H.E. Stanley, PhysioBank, PhysioToolkit, and PhysioNet: components of a new research resource for complex physiologic signals, Circulation 101 (23) (2000) 215–220.

[25] P. Kligfield, L.S. Gettes, J.J. Bailey, R. Childers, B.J. Deal, E. William Hancock, G. Van Herpen, J.A. Kors, P. Macfarlane, D.M. Mirvis, et al., Recommendations for the standardization and interpretation of the electrocardiogram, Circulation 115 (10) (2007) 1306–1324.