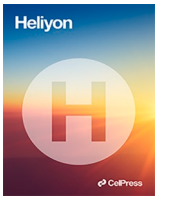
Heliyon 9 (2023) e16297

Contents lists available at ScienceDirect

Heliyon

journal homepage: www.cell.com/heliyon



Research article

Smart scan of medical device displays to integrate with a mHealth application

Pedro Lobo [a,b,*], João L. Vilaça [a,b], Helena Torres [a,b,c,d,e], Bruno Oliveira [a,b,c,d,e], Alberto Simões [a,b]

[a] 2AI, School of Technology, IPCA, Barcelos, Portugal
[b] LASI – Associate Laboratory of Intelligent Systems, Guimarães, Portugal
[c] Algoritmi Center, School of Engineering, University of Minho, Guimarães, Portugal
[d] Life and Health Sciences Research Institute (ICVS), School of Medicine, University of Minho, Braga, Portugal
[e] ICVS/3B's -PT Government Associate Laboratory, Braga/Guimarães, Portugal

ARTICLE INFO

Keywords:
mHealth
AI (Artificial intelligence)
OCR (Optical character recognition)
Medical devices
Data aggregation
Consumer health information
Data collection
e-health

ABSTRACT

Background: The daily monitoring of the physiological parameters is essential for monitoring health condition and to prevent health problems. This is possible due to the democratization of numerous types of medical devices and promoted by the interconnection between these and smartphones. Nevertheless, medical devices that connect to smartphones are typically limited to manufacturers applications.
Objectives: This paper proposes an intelligent scanning system to simplify the collection of data displayed on different medical devices screens, recognizing the values, and optionally integrating them, through open protocols, with centralized databases.
Methods: To develop this system, a dataset comprising 1614 images of medical devices was created, obtained from manufacturer catalogs, photographs and other public datasets. Then, three object detector algorithms (yolov3, Single-Shot Detector [SSD] 320 × 320 and SSD 640 × 640) were trained to detect digits and acronyms/units of measurements presented by medical devices. These models were tested under 3 different conditions to detect digits and acronyms/units as a single object (single label), digits and acronyms/units as independent objects (two labels), and digits and acronyms/units individually (fifteen labels). Models trained for single and two labels were completed with a convolutional neural network (CNN) to identify the detected objects. To group the recognized digits, a condition-tree based strategy on density spatial clustering was used.
Results: The most promising approach was the use of the SSD 640 × 640 for fifteen labels.
Conclusion: Lastly, as future work, it is intended to convert this system to a mobile environment to accelerate and streamline the process of inserting data into mobile health (mhealth) applications.

1. Introduction

Daily health check-ups play an important role in the early prevention of health problems. In the health sector, different hospitals

* Corresponding author. 2AI, School of Technology, IPCA, Barcelos, Portugal.
E-mail addresses: pjlobo@ipca.pt (P. Lobo), jvilaca@ipca.pt (J.L. Vilaça), htorres@ipca.pt (H. Torres), boliveira@ipca.pt (B. Oliveira), asimoes@ipca.pt (A. Simões).

https://doi.org/10.1016/j.heliyon.2023.e16297
Received 4 April 2023; Received in revised form 11 May 2023; Accepted 12 May 2023
Available online 29 May 2023

and governmental organizations are developing mobile health (mhealth) applications to centralize information about patients. Some of this information can be filled in by the patients, in their everyday life, like their blood pressure or glycemic levels.

Currently, to record medical information from these applications, the patient needs to enter manually the obtained data. While some devices offer wireless and/or Bluetooth connectivity, they are limited to be used by the manufacturers' own applications, using proprietary protocols. Data entry is one of the most mentioned features of mhealth applications for diabetes and hypertension in user reviews [1].

The work developed and presented in this paper aims to automate the digitization of the information presented by medical devices, aggregating this data from different sources in a single application, through the development of a smart scan system. This consisted in the development of a framework that results from the combination of different architectures that were tested as will be presented throughout this paper. This system is directed to mobile applications so that patients will be able to use the camera of their smartphone to read the measurements presented on screens of medical devices as well as the identification of the device itself, allowing the automatic registering of values on a local database, or sent to a centralized system. In this way, the relevance of this study is not just about applying Artificial Intelligence (AI) to medical data, but about developing a framework that will positively impact people's health care in their daily lives [2].

The idea of developing a smart scan of medical device display was already explored in some previous works. Finnegan et al. used traditional image processing methods to identify regions of interest containing segments of the seven-segment, and then classified the digits using a MLP classifier [3]. Their system achieved a classification accuracy of 93% and an F1 score of approximately 84. The framework that we propose aims to improve the accuracy by using deep learning methods instead of traditional image processing methods to detect digits.

Another work that proposes a solution on this topic was developed by Tsiktsiris et al. [4]. The project describes the development of a scanning system for medical device screens to help elderly people read measurements using a smartphone, which involved adaptive thresholding, segmentation, and classification using a decision tree. The system achieved an accuracy of 96.22% for blood pressure meters but was limited to only that device. The framework that we propose in the current paper aims to expand the range of devices and develop a versatile scanning system that is not dependent on the contrast between digits/background.

Shenoy and Aalami, created a mobile application for patients to scan medical measurements to later be reported to a doctor [5]. The interface of the application allows the user to select the type of medical device to be scanned, and the device's camera is used to fit the digits within a predefined bounding box for analysis. A back-end server is used to process the information, requiring an active connection. To process the analyses, the image is divided into individual digits, and each digit is classified using the Random Forest Classifier, which was chosen due to its accuracy of 98.2%. Unlike Shenoy and Aalami work, we intend to develop a solution that automatically detects and analyzes them and runs locally on the mobile device.

In the industrial sector, there is also a need to digitize sampled digits on displays, and Kulkarni and Kute proposed a new method for reading seven-segment digits in images [6]. They focused on devices with a backlight, making it easier to distinguish the digits from the background through binarization and simple filters. After rotating and segmenting the image, they used a decision tree based on the distribution of black data pixels for classification. Their method achieved an accuracy of 79%.

Through our study, five types of devices were explored: glycemia and blood pressure monitors, whose measurements are typically collected in current mhealth applications; thermometers, oximeters, and scales, which are common devices to be found in most people's homes. Given oximeters and blood pressure monitors also measure the patient's heartbeat, the system can read and store these values as well.

In the case of oximeters, glycemia, and blood pressure meters, the scanning process consists of reading the digits of the measurements and the detection of the abbreviations or measurement units shown on the device. This information will be used to, automatically, detect the type of device and, when relevant, the measurement unit. For scales and thermometers, due to the lack of acronyms or unit of measurement in many cases, or the small size of this information, the scanning process was developed to only identify the digits and not the device itself. Therefore, users must manually indicate the type of device being used.

The main contributions of this paper are:

- A dataset of photos of medical device screens for developing machine learning and object detection techniques;
- Comparison of performance of state-of-the-art strategies for object detection (SSD and Yolo) applied to the detection of digits and abbreviations/measurement units present on the display of the devices;
- Development of a condition tree to differentiate/classify the data presented on medical devices screens by medical analysis;

The remaining of this chapter is organized as follows: The framework and the description of the procedures necessary for the implementation of the methods is made in Section 2. In Section 3, the results of the studied methods are presented. After all the experiences, Section 4 performs an analysis of the results obtained. Finally, we conclude with some final remarks in Section 5.

2. Methodology

2.1. Overview

As depicted in Fig. 1, the proposed method relies on four conceptual blocks:

1. The creation of two distinct datasets: one for the object detection task (digits and other features, like unit labels), and another one, extracted from the first, for the classification of the detected objects. The first dataset was artificially augmented to increase the number of images and to add variability and noise.
2. The training of object detection models using the collected dataset. This model is responsible for the generation of the bounding box containing digits, measure units and/or acronyms usual in certain medical devices.
3. After detecting the position of the objects, another model is trained to classify each area, either with the correct detected digit, or the measurement units in the display of the medical device.
4. Finally, a condition tree strategy is used to group the detected digits, forming values. This process takes information about which device is being analyzed, as well as the position of the digit.

2.2. Datasets

This section describes the two datasets preparation: the object detection dataset (to detect the places where digits and/or measurement units are shown), and the second dataset, used to classify these regions as each one of the ten available digits or the measurement different units.

2.2.1. Object detection dataset

The object detection dataset was constructed by taking pictures of some local devices, the compilation of images of medical devices found on the internet, namely from catalogues from manufacturers, as well as including the images present in a public dataset [3]. This dataset is characterized in Table 1 (it includes the number of images obtained from the public dataset [3], as well as counts per medical device oximeter, glucometer, and blood pressure monitor).

Each image was subject to three steps:

- Preprocessing: Given the images were obtained from different sources, the first step normalizes them, namely guaranteeing a common file format and image size (832 $\times$ 832 pixels).
- Data augmentation: To create diversity the original dataset was subject to data augmentation[1] (offline). This step increased the number of images in the dataset, from 1614 images up to 30,431 images. These new images add variability and noise, to allow a more robust model, less sensible to structural changes of the objects detected through the transformations of crop, scale, shear, translation, and rotation. To give robustness to different types of lighting and noise, nearest neighbor, bilinear interpolation, Gaussian/average/media blur, color channels inversion, brightness, hue, and saturation changes, contrast normalization and grayscale transformations were also applied.
- Labelling: This dataset was manually labelled, describing the bounding boxes for each interesting object and, for each object, the class to which it belongs. Thus, a fifteen-label list was created, including the ten digits (from 0 to 9), and five units/acronyms labels: pulse, oxygen level, glycemic level, and systolic and diastolic blood pressures. Note that, through this process, it was possible to generate the labeling dataset for two distinct classes (digits or acronyms/units) and for fifteen classes (thus, distinguishing between the ten digits and the five different acronyms/units).

2.2.2. Object classification dataset

From the areas of the abovementioned dataset, another dataset was constructed for object classification. To extract the images, was decided to use a YOLOv3 model trained over the original dataset (22,860 images for training, and 7571 images for testing). The bounding boxes generated by the model led to a dataset of 495,647 images. These images were assigned to one of the fifteen classes already described and scaled to a dimension of 32 $\times$ 32 pixels.

An alternative approach could reuse the manual labeling present in the original dataset. However, the manual labeling bounding boxes are perfect and may not give the necessary robustness to the Convolutional Neural Network used in the classifier.

Table 2 present some examples of images for the five measurement possibilities, and Table 3 present the number of images for each class. This table also presents the distribution of each label for the training and test phases.

2.3. System framework

This section describes the system framework. It starts with the object detection model, with a description of the attempted experiments as well as the selected approach. Afterward, the classifier model is presented, where the full neural network structure will be depicted. Finally, the condition tree for digit clustering will be explained.

2.3.1. Object detector

Different object detection models were attempted, to extract the information present on the screens of the medical devices. Three distinct approaches were tested: YOLOv3 [7] and SSD Mobilenet v2 [8] (using both 320 $\times$ 320 and 640 $\times$ 640).[2]. These models were selected given that the intelligent scanning system being developed is intended to run on smartphones, so they should be less

[1] https://github.com/mukopikmin/bounding-box-augmentation (2021).
[2] https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf2_detection_zoo.md (2021).

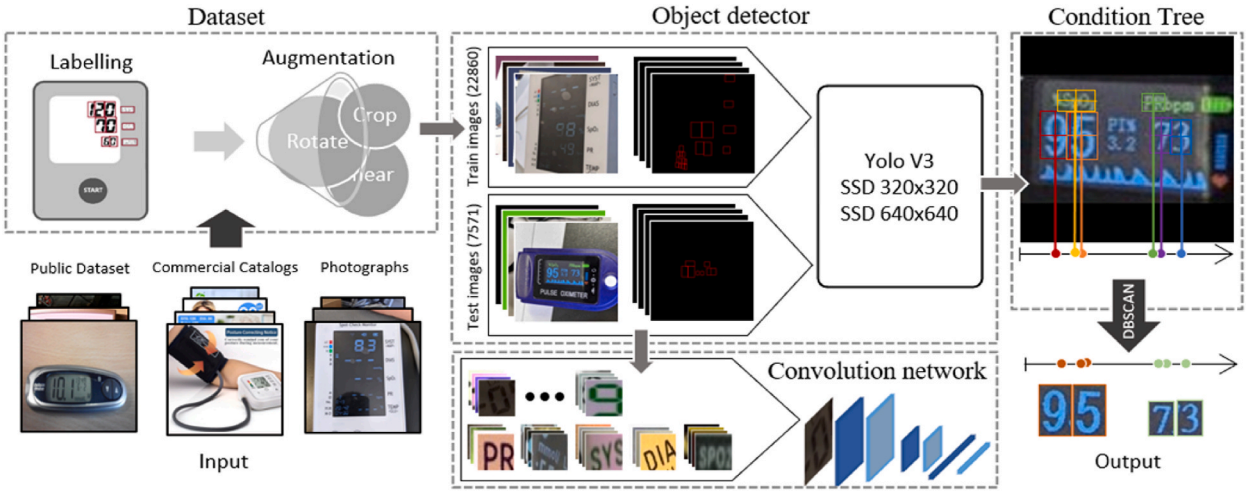


Fig. 1. Overview of the proposed smart scan system.

Table 1
Number of images and devices from each source.

	Commercial Catalogs		Photographs		Public Dataset	
	Nr of Dev.	Nr of Img.	Nr of Dev.	Nr of Img.	Nr of Dev.	Nr of Img.
Oximeter	~23	253	1	68	0	0
Glucometer	~39	233	0	0	3	254
Blood Pressure Monitor	~57	307	1	111	3	388

Table 2
Examples of images for each of the five acronyms/units.

Pulse	Glycemic Level	Systolic Pressure	Diastolic Pressure	Oxygen Saturation
				

Table 3
Distribution of images by label.

Label	Dataset		Train Set		Test Set	
Zero	52,949	10.7%	42,432	10.5%	10,517	11.7%
One	71,658	14.5%	57,769	14.2%	13,889	15.4%
Two	50,348	10.2%	41,583	10.3%	8765	9.8%
Three	27,205	5.5%	22,647	5.6%	4558	5.1%
Four	26,920	5.4%	21,852	5.4%	5068	5.6%
Five	31,571	6.4%	26,132	6.4%	5439	6.1%
Six	35,543	7.2%	29,260	7.2%	6283	6.9%
Seven	33,354	6.7%	27,106	6.7%	6248	6.9%
Eight	60,473	12.2%	50,724	12.5%	9749	10.8%
Nine	37,418	7.6%	31,118	7.7%	6300	7.0%
Pulse	18,804	3.8%	15,448	3.8%	3356	3.7%
Glycemic	8775	1.8%	7213	1.8%	1562	1.7%
Systolic	14,959	3.0%	11,602	2.9%	3357	3.7%
Diastolic	16,577	3.3%	13,077	3.2%	3500	3.9%
SpO_2	9093	1.8%	7751	1.9%	1342	1.5%
Total:	495,647		405,714		89,933	

computationally expensive as possible. This is also the reason why the object detector models were tried using single-stage and not two-stage [9].

To test the different models was used a workstation with a NVidia GTX 1070 8 GB, an Intel Core i7-8700, and 32 GB of RAM, running with the NVIDIA libraries CUDA 11.0 and CUDNN 8.0.4 for YOLOv3, and CUDNN 8.1.0 for the SSDs models.

Regarding the network structures, for YOLOv3 was used the Darknet (darknet53.conv.74[3]) and, in the case of SSDs was used in the TensorFlow Model Garden API (v2). These models were trained not just to compare their performance but also to verify under which conditions they have the best performance. For that, all models were trained with three different goals:

- a model trained to detect a single object, where digits and acronyms/units are the same label;
- a model trained to detect and classify objects in two distinct classes: digits or acronyms/units;
- a model trained to detect and classify objects in fifteen classes (thus, distinguishing between the ten digits and the five different acronyms/units).

All training processes were performed until the loss got stable. Table 4 presents the number of training steps and the obtained loss for each of the nine combinations of models.

2.3.2. Classifier

To perform the classification of the detected objects was used a Convolutional Neural Network (CNN). This choice was based on Kulkarni and Kute previous work, where this approach presented better results in similar contexts when compared to other classifiers [10].

This solution is used together with the object detector models described before, which were trained for single label. The classifier was trained to distinguish between the fifteen classes described for the second dataset. The architecture for the CNN was based on the work of Dimitrios Roussis[4] and is summarized on Table 5. The training process ran for 200 epochs of 100 steps each.

2.3.3. Condition tree

After the detection of the digits, they need to be grouped to make up the measurement's values. To perform this operation was used a clustering condition tree, based on the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) technique, as it has a non-fixed number of labels to group the measurements. The data used as input for this process came from the object detector step and consisted of the x and y coordinates referent to the centroids of the detected information.

(A) Pre-processing: The pre-processing consists of two steps: filtering data that do not represent any medical analysis, and forming the cluster's input. Some devices show some other information that are not relevant (for instance, the number identifier of the patient, for devices that allow to track measures for more than one user). These digits are easily filtered because they have a smaller occupied area compared to the relevant values. DBSCAN requires setting a density threshold. As the scanning distance is variable (between camera and medical device screen), the average proximity between detected objects also varies so that a fixed threshold value cannot be applied to all situations. The solution was to identify the limits (maximum and minimum) of the detected bounding boxes information and crop this part of the image. Then, this area of interest is normalized to a fixed resolution of 1:1 without being deformed by adding vertical or horizontal edges if necessary. The transformation applied to the original image is also applied to the coordinates of the centroids. (b) Clustering: Devices were divided into 2 groups: those that only display one measure (scales, glucometers and thermometers) and those that display more than one measure on the screen (such as oximeters —oximetry and pulse—, and blood pressure monitors —systolic, diastolic and sometimes pulse).

As previously stated, sometimes the devices have further information not relevant for extraction, that are easily filtered out given their occupation area on the screen. Thus, after this filtering, it is easy to extract the relevant information for the first type of device, without applying any clustering technique.

For the second group of devices, the clustering process is responsible for relating obtained digits accordingly with their spatial position. This information was based on the proximity of the centroids of the bounding boxes of the objects detected during the object detection step. One question that arose during the development was whether it would be more efficient to perform the clustering using 2D coordinates (x and y centroid coordinates) or 1D coordinates (x or y centroid coordinates).

When processing data in 2D perspective, the cluster application works in the same way for all devices in the second group. However, in a 1D perspective, it is necessary to pay attention to the devices under analysis, if all the measures present on the screen of a blood pressure monitor are horizontally aligned, or, in the case of the oximeter, they may be aligned horizontally or vertically. Fig. 2 explains this situation.

At first, the 2D perspective seemed to be better, as it is compatible with all devices, however, a problem was found that is directly related to the 7-segment digit screens. On these screens, the digits are displayed in specific places, so there is no adjustment for the distance between them. In this way, the digit '1' appears aligned to the right, presenting a greater distance in relation to the digits to its left and smaller to the right. On the other hand, the acronyms are sometimes far from the digits corresponding to the same value, despite being aligned horizontally or vertically.

[3] https://pjreddie.com/darknet/yolo/(2021).
[4] https://www.kaggle.com/code/dimitriosroussis/svhn-classification-with-cnn-keras-96-acc/notebook (2021).

Table 4
Training steps and final average loss.

	1 Label		2 Labels		15 Labels	
	Steps	Loss	Steps	Loss	Steps	Loss
YOLOv3	4k	1284	4k	1328	21k	1632
SSD 320 × 320	50k	0,340	50k	0,344	50k	0,424
SSD 640 × 640	58k	0,293	56k	0,291	56k	0,332

Tables 5
15 class CNN Classifier Architecture.

Layer type	Output Shape	Param. Count
Reshape	(32, 32, 1)	0
Batch Normalization	(32, 32, 1)	4
Conv2D	(32, 32, 1)	320
Max Pooling 2D	(16, 16, 32)	0
Dropout	(16, 16, 32)	0
Conv2D 1	(16, 16, 64)	18,496
Batch Normalization 1	(16, 16, 64)	256
Conv2D 2	(16, 16, 64)	36,928
Max Pooling 2D 1	(8, 8, 64)	0
Dropout 1	(8, 8, 64)	0
Conv2D 3	(8, 8, 128)	73,856
Batch Normalization 2	(8, 8, 128)	512
Conv2D 4	(8, 8, 128)	147,584
Max Pooling2D 2	(4, 4, 128)	0
Drop out 2	(4, 4, 128)	0
Flatten	(2048)	0
Dense	(128)	262,272
Drop out 3	(128)	0
Dense 1	(15)	1935
Total parameters: 542,163		
Trainable parameters: 541,777		
Non-trainable parameters: 386		

3. Results

3.1. Object detectors

As previously mentioned, the models were tested in three different situations: one label object detection, 2 label object detection (numbers or units) and 15 labels detection (each digit and unit, individually). The performance measures, per object, for the YOLOv3, SDD 320 × 320 and SSD 640 × 640 models are shown in the Tables Tables 6 and 7, respectively, and the total are shown in the Tables Tables 8, 9 and 10. These measures were computed with an Intersection over Union (IoU or Jaccard Index) average threshold of 0.5.

The SSD 320 × 320 model stands out from the rest given the low values for recall and accuracy, resulting in a low F1- score. Regarding YOLOv3 and SDD 640 × 640, are the ones with more comparable results but with YOLOv3 having more consistent across all metrics.

3.2. Classifier

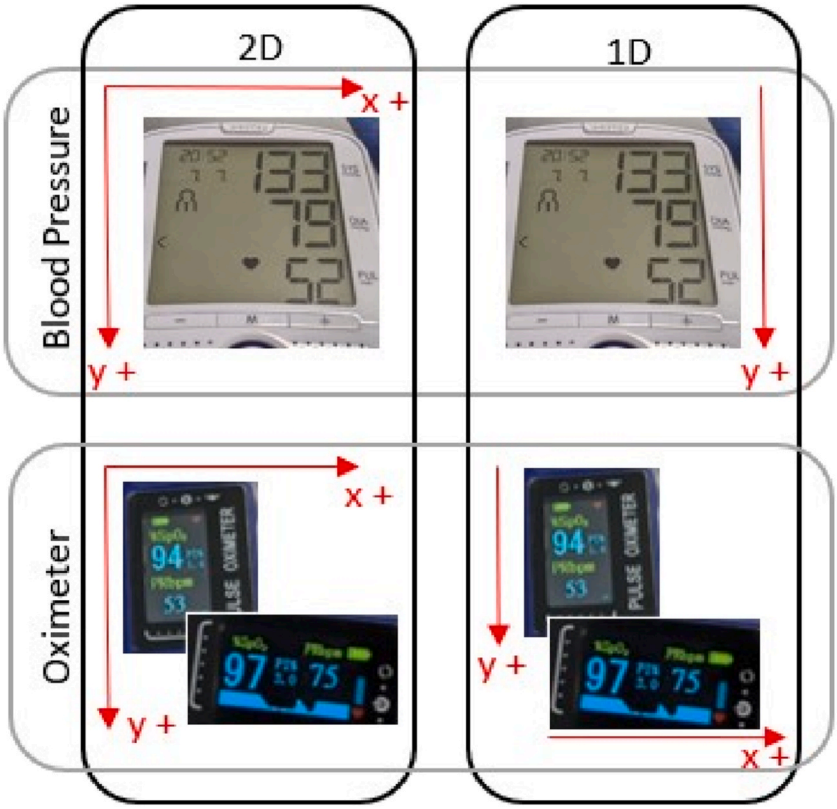
Regarding the classifier, Fig. 3 shows the loss and accuracy evolution during this model training. To make it fit on a mobile device, the model was converted to TFlite and quantized. After this process, a 75% lighter model was obtained (from 2123 KBs to 548 KBs) with an accuracy of 96.99%.

3.3. Object detector with classifier

This topic presents the results of the object detection models, trained for 1 and 2 labels, in series with the classifier. Although the two methods presented good results individually, in series we obtained worse results, as can be seen in Tables 11 and 12, per object, and total in Tables 13 and 14.

4. Discussion

Regarding the three object detection models as well as the three different conditions in which they were tested, it allowed to obtain



Figs. 2. 1D and 2D cluster.

Table 6
Accuracy and recall, per object, of object detection models for 2 labels (0.50IoU).

	SSD 320 × 320		SSD 640 × 640		YOLOv3	
	Precision	Recall	Precision	Recall	Precision	Recall
Digits	0,9080	0,5017	0,9723	0,7300	0,9397	0,8852
Acronyms & Units	0,9000	0,2722	0,9501	0,6345	0,9130	0,8365

Table 7
Accuracy and Recall, per object, of the Object Detection Models for 15 labels (0.50IoU).

	SSD 320 × 320		SSD 640 × 640		YOLOv3	
	Precision	Recall	Precision	Recall	Precision	Recall
Zero	0,9650	0,3252	0,9705	0,6679	0,8631	0,8925
One	0,9458	0,2861	0,9724	0,5893	0,9129	0,8754
Two	0,9272	0,3373	0,9169	0,6548	0,8996	0,8938
Three	0,9787	0,4323	0,9701	0,7159	0,9346	0,8919
Four	0,9710	0,6055	0,9694	0,8528	0,9430	0,9542
Five	0,9217	0,4422	0,9163	0,7423	0,9263	0,9003
Six	0,9694	0,5183	0,9624	0,7710	0,9356	0,8927
Seven	0,9716	0,5066	0,9776	0,7668	0,9554	0,8855
Eight	0,9617	0,3837	0,9558	0,6896	0,9010	0,8712
Nine	0,9640	0,4709	0,9627	0,7667	0,9295	0,9046
Pulse	0,8994	0,0879	0,9475	0,5487	0,8663	0,9300
Glycemic	0,9586	0,4908	0,9550	0,7207	0,9437	0,8897
Systolic	0,9304	0,2172	0.9662	0,6450	0,9227	0,9371
Diastolic	0,9280	0,1899	0,9658	0,6102	0,9122	0,9474
SpO2	0,9752	0,2730	0,9270	0,7920	0,8871	0,8986

Table 8
Object detector model for 1 label (0,50IoU).

	SSD 320 × 320	SSD 640 × 640	YOLOv3
Precision	0,9011	0,9710	0,9450
Recall	0,4709	0,7086	0,8748
F1-score	0,6186	0,8193	0,9085
Accuracy	0,4478	0,6940	0,8324

Table 9
Object detector model for 2 label (0,50IoU).

	SSD 320 × 320	SSD 640 × 640	YOLOv3
Precision	0,9073	0,9691	0,9356
Recall	0,4659	0,7152	0,8777
F1-score	0,6157	0,8230	0,9057
Accuracy	0,4448	0,6993	0,8277

Table 10
Object detector model for 15 label (0,50IoU).

	SSD 320 × 320	SSD 640 × 640	YOLOv3
Precision	0,9555	0,9639	0,9140
Recall	0,3386	0,6887	0,8993
F1-score	0,5000	0,8034	0,9066
Accuracy	0,3350	0,6760	0,8425

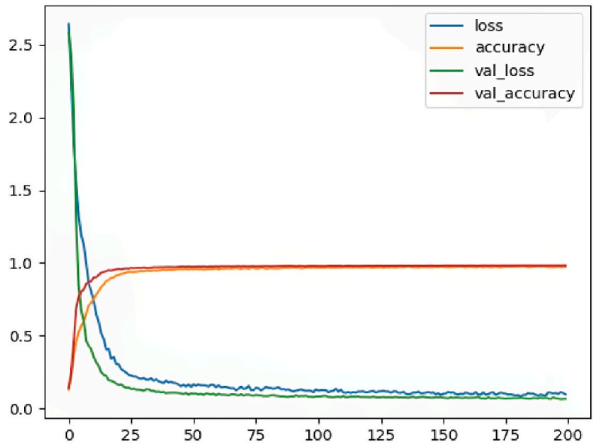


Fig. 3. Loss and accuracy evaluation of the classifier training.

interesting results. It is necessary from now to highlight the SSD 320 × 320 for the poor results compared to the others. Looking at the architecture of this model, the input resolution of the images is lower than that of the rest, so the images may have lost some information after resizing.

Regarding the three conditions in which the models were trained, for single label, double label and fifteen labels, no significant differences were detected in their performance. The models were able to handle numerous objects with different characteristics for the same class as well as a greater number of classes that they had to identify.

On the other hand, the results of the object detectors in series with the classifier were less than expected. It is thought that the small irregularities and shifts of the bounding boxes generated by the object detection models may have significantly influenced the CNN results.

To choose the object detection model, it was necessary to be aware that the project we are developing has as its focus running on mobile devices. The models cannot remain in the raw state as they are in after training. These need to be compressed to run on this type of device, which consists of converting to ".tflite" format as well as metadata indexing (model information, inputs and outputs such as tag mapping). SSD models offer greater compatibility in this transfer to a mobile environment compared to YOLOv3. By analyzing the results, despite the YOLOv3 presenting a better performance, the choice of the final model that would be integrated in the application was the SSD 640 × 640 for fifteen labels. We think that the difference between the performance of the two models does not justify the

Table 11
Accuracy and Recall, per object, of the Object Detector Model for 1 Label with Classifier (0,50IoU).

	SSD 320 × 320		SSD 640 × 640		YOLOv3	
	Precision	Recall	Precision	Recall	Precision	Recall
Zero	0,8606	0,3641	0,9364	0,6081	0,8899	0,6509
One	0,7462	0,2468	0,9005	0,4091	0,8557	0,4937
Two	0,9237	0,3678	0,9673	0,5743	0,9642	0,6303
Three	0,8285	0,4609	0,9082	0,6525	0,8379	0,7197
Four	0,9537	0,4024	0,9869	0,5202	0,9691	0,5355
Five	0,9302	0,4839	0,9686	0,7026	0,9542	0,7343
Six	0,9625	0,5062	0,9651	0,6629	0,9713	0,6735
Seven	0,8662	0,4233	0,9561	0,5780	0,9225	0,5397
Eight	0,3646	0,4904	0,4034	0,7050	0,3193	0,8519
Nine	0,7054	0,4992	0,7696	0,6917	0,6952	0,7526
Pulse	0,8011	0,0928	0,9398	0,2829	0,8546	0,3413
Glycemic	0,6153	0,3868	0,7168	0,5424	0,6159	0,6372
Systolic	0,9085	0,2031	0,9434	0,4660	08889	0,5539
Diastolic	0,7103	0,2259	0,8060	0,5233	0,6770	0,6622
SpO2	0,4172	0,3033	0,5385	0,6360	0,3105	0,5685

Table 12
Accuracy and Recall, per object, of the Object Detector Model for 2 Labels with Classifier (0,50IoU).

	SSD 320 × 320		SSD 640 × 640		YOLOv3	
	Precision	Recall	Precision	Recall	Precision	Recall
Zero	0,8645	0,3584	0,9300	0,6044	0,8797	0,6515
One	0,7536	0,2426	0,8896	0,4112	0,8249	0,4750
Two	0,9357	0,3689	0,9642	0,5750	0,9626	0,6245
Three	0,8269	0,4642	0,9055	0,6517	0,8376	0,7197
Four	0,9561	0,3982	0,9834	0,5137	0,9602	0,5367
Five	0,9301	0,4806	0,9681	0,7039	0,9526	0,7302
Six	0,9584	0,5062	0,9717	0,6647	0,9703	0,6567
Seven	0,8780	0,4199	0,9543	0,5841	0,9055	0,5265
Eight	0,3756	0,4880	0,4005	0,7092	0,3054	0,8501
Nine	07186	0,4907	0,7625	0,6936	0,6825	0,7422
Pulse	0,8261	0,0934	0,9270	0,3005	0,8472	0,3355
Glycemic	0,6284	0,3946	0,7066	0,5552	0,5890	0,6061
Systolic	0,9092	0,2049	0,9351	0,4723	0,8834	0,5469
Diastolic	0,7324	0,2187	0,8024	0,5384	0,6603	0,6396
SpO2	0,3893	0,2652	0,5517	0,6516	0,2903	0,5624

Table 13
Object detector model for 1 label with classifier (0,50IoU).

	SSD 320 × 320	SSD 640 × 640	YOLOv3
Precision	0,7214	0,7808	0,6596
Recall	0,3742	0,5721	0,6120
F1-score	0,4928	0,6604	0,6349
Accuracy	0,3552	0,5598	0,5831

Table 14
Object detector model for 2 label with classifier (0,50IoU).

	SSD 320 × 320	SSD 640 × 640	YOLOv3
Precision	0,7798	0,7800	0,7021
Recall	0,5769	0,5769	0,6608
F1-score	0,6632	0,6632	0,6808
Accuracy	0,5635	0,5636	0,6209

agility which the SSD 640 × 640 offers when being transported to a mobile environment.

5. Conclusion

In this paper we presented a framework to detect and recognize measures from five different medical devices: glucometers, scales, thermometers, blood pressure monitors, and oximeters.

The developed dataset is not made completely public given it contains images from company catalogs, that have copyright issues. Nevertheless, the authors would be happy to share it with other researchers, when contacted directly.

Furthermore, all the methods that make up the proposed intelligent scanning system were successfully validated. In this way, it was possible to determine the best technologies that will integrate the Mobile Application Library for Smart Scan of Medical Device Displays that will be developed. As future work, there is room to refine the methods covered as well as the dataset, however the added value will consist of transporting the system developed in this paper to a mobile environment, as well as testing it by a group of volunteers.

Author contribution statement

Pedro Lobo: Conceived and designed the experiments; Performed the experiments; Analyzed and interpreted the data; Contributed reagents, materials, analysis tools or data; Wrote the paper. João L. Vilaça, Helena Torres, Bruno Oliveira, Alberto Simões: Conceived and designed the experiments; Wrote the paper.

Funding statement

Pedro Lobo was supported by Northern Portugal Regional Operational Programme {NORTE-01-0145-FEDER-000045; NORTE-01-0145-FEDER-000059}, Fundação para a Ciência e a Tecnologia {UIDB/05549/2020; UIDP/05549/2020; CEECINST/00039/2021; LASI-LA/P/0104/2020; SFRH/BD/136721/2018; SFRH/BD/136670/2018; COVID/BD/154329/2023; COVID/BD/154328/2023}, Next Generation EU {HfFP – Health From Portugal}.

Data availability statement

Data will be made available on request.

Declaration of competing interest

Pedro Lobo reports financial support was provided by Fundação para a Ciência e Tecnologia, Northern Portugal Regional Operational Programme, Next Generation EU.

Acknowledgments

This work was funded by the projects “NORTE-01-0145-FEDER-000045” and “NORTE-01-0145-FEDER-000059”, supported by Northern Portugal Regional Operational Programme (NORTE 2020), under the Portugal 2020 Partnership Agreement, through the European Regional Development Fund (FEDER). It was also funded by national funds, through the FCT (Fundação para a Ciência e a Tecnologia) and FCT/MCTES in the scope of the project UIDB/05549/2020, UIDP/05549/2020, CEECINST/00039/2021 and LASI-LA/P/0104/2020. This project was also funded by the Innovation Pact HfFP – Health From Portugal, co-funded from the “Mobilizing Agendas for Business Innovation” of the “Next Generation EU” program of Component 5 of the Recovery and Resilience Plan (RRP), concerning “Capitalization and Business Innovation”, under the Regulation of the Incentive System “Agendas for Business Innovation”. The authors also acknowledge FCT, Portugal and the European Social Found, European Union, for funding support through the “Programa Operacional Capital Humano” (POCH) in the scope of the PhD grants SFRH/BD/136721/2018, COVID/BD/154329/2023 (Oliveira B.) and SFRH/BD/136670, COVID/BD/154328/2023 (Torres H. R.).

References

[1] S. Wang, H.S. Lee, W. Choi, A feature-oriented analysis of developers’ descriptions and user reviews of top mHealth applications for diabetes and hypertension, Int. J. Med. Inf. 156 (Dec. 2021) 104598.
[2] F. Cabitza, A. Campagner, The need to separate the wheat from the chaff in medical informatics: introducing a comprehensive checklist for the (self)-assessment of medical AI studies, Int. J. Med. Inf. 153 (Sep. 2021) 104510.
[3] E. Finnegan, M. Villarroel, C. Velardo, L. Tarassenko, Automated method for detecting and reading seven-segment digits from images of blood glucose metres and blood pressure monitors, J. Med. Eng. Technol. 43 (6) (Aug. 2019) 341–355.
[4] D. Tsiktsiris, K. Kechagias, M. Dasygenis, P. Angelidis, Accelerated seven segment optical character recognition algorithm, in: 5th Panhellenic Conference on Electronics and Telecommunications, PACET, 2019. Nov. 2019.
[5] V.N. Shenoy, O.O. Aalami, Utilizing smartphone-based machine learning in medical monitor data collection: seven segment digit recognition, AMIA Annu. Symp. Proc. 2017 (2017) 1564. Accessed: Aug. 31, 2021. [Online]. Available:/pmc/articles/PMC5977613/.
[6] P.H. Kulkarni, P.D. Kute, Optical numeral recognition algorithm for seven segment display, in: Conference on Advances in Signal Processing, CASP, 2016, pp. 397–401. Nov. 2016.
[7] J. Redmon, A. Farhadi, YOLOv3: an Incremental Improvement, Apr. 2018.

[8] W. Liu, et al., SSD: single shot MultiBox detector, in: Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) vol. 9905, LNCS, Dec. 2015, pp. 21–37.
[9] P. Soviany, R.T. Ionescu, Optimizing the trade-off between single-stage and two-stage deep object detectors using image difficulty prediction, in: Proceedings - 2018 20th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing, SYNASC, 2018, pp. 209–214. Sep. 2018.
[10] W. Liu, J. Wei, Q. Meng, Comparisions on KNN, SVM, BP and the CNN for handwritten digit recognition, in: Proceedings of 2020 IEEE International Conference on Advances in Electrical Engineering and Computer Applications, AEECA, Aug. 2020, pp. 587–590.