# The Vital Extraction Challenge

## Final Report, *CloudPhysician* Mid-prep Problem statement


## Abstract

Through this work, we try to bridge the gap between patients, nurse and doctors. We have proposed a model that can extract the information about a patient's vitals such as heart rate, blood pressure, pulse rate, etc. from the monitor screen. This would enable the doctors to analyze the patients' conditions and accordingly take the necessary actions. The model was trained using yolov7 in a Google Colab notebook. To ensure accurate results, pre-processing had to be done for the data proided in the form of images and data files. Our model is good to be used as apart of the Smart ICU solution to help doctors monitor multiple patients in real-time right from their desk.


## Introduction

In this problem statement, we had to detect the information about a patient's vitals from the images containing a snapshot of the vital monitoring screen. Our goal was to extract the monitor screen from the input image (or remove the background to focus on the monitor screen) and locate and interpret the vitals. The targeted vital metrics were heart rate, spO2, Respiration rate(RR), Systolic and Diastolic blood pressure, and MAP(Mean Arterial Pressure). Our approach to this problem was a series of routines, including deep learning, computer vision, and semi-supervised Machine Learning.

We ensured that the input fed to our pipeline was relevant, unbiased, and consistent with the model type at each pipeline stage. The solution can be broken down into three significant steps. In step one, we extract the monitor screen from the image containing the monitor screen with some background. To execute this, we trained the Monitor Segmentation dataset on Yolo, an object-detection, deep learning algorithm. Yolo outperforms many object-detection architectures because of its fast computation, generalization of the detected objects, and contextual interpretation of the inputs.

The next step was to classify the monitor screen(into one of the four classes, as provided) and extract the features. We employed Yolo on the input prepared from the Classification dataset. In this step, we detected the objects corresponding to the vitals. The objects included textual and graphical information.

We used Easy OCR to interpret the text objects. Easy OCR is a fast, flexible, and reliable font/character interpreter. Through this, we extracted the heart rate, blood pressure, and other relevant metrics as numbers.

It is crucial to interpret various graphs on the monitor screen to comprehend a patient's condition thoroughly. We designed a graph digitization stage in our solution pipeline to operationalize this. It was a series of image preprocessing and clustering algorithms. We used thresholding methods to create detectable distinctions between black and non-black pixels in the input image. Subsequently, we applied K-means and DBSCAN clustering algorithms to extract meaningful information from the graphical objects.

We experimented with various distinct models and architectures to design this pipeline. This report describes all the stages of the pipeline in exhaustive detail. We hope that our solution will be useful in overcoming similar challenges.

## Dataset

### Data Analysis and Preprocessing

There were three main datasets provided to us -

1. Monitor Dataset
2. Classification Dataset
3. Unlabelled Dataset

**Monitor Dataset** It consisted of a total of 2000 images of different types of monitors with some amount of background and captured from various angles. These images had to be processed in such a way that the cropped images contain only the main monitor screen. The monitors were to be classified into 4 different sets such that, for any random input image, the model would be able to group it into one of these classes.

The dataset is given in the form of images and a label file corresponding to those images. The label file contains the co-ordinates for creating a quadrilateral to crop the monitor screen from the image. The coordinates are given as [x1, y1, x2, y2, x3, y3, x4, y4], where x1 is the pixel distance from left end of image and y1 is the pixel distance from top of the

image. The size of the images given in this dataset is 1280 x 720.

| | A | B |
|---|---|---|
| 1 | image_name | points |
| 2 | medicakolkata_mau | [459.5, 169.5, 886.0 |
| 3 | kamalnayanbajaj_m | [422.4, 158, 967.8, 1 |
| 4 | kamalnayanbajaj_m | [497.4, 184.9, 792.7 |
| 5 | medicakolkata_ccu2 | [353.799987792968 |
| 6 | medicakolkata_ccu2 | [246.9, 155.3, 756.5 |
| 7 | hcgbhavnagar_micu | [228.9, 110.4, 808.4 |
| 8 | medicakolkata_ccu1 | [467.4, 188.1, 854.1 |
| 9 | medicakolkata_ccu2 | [337, 165.80000305 |
| 10 | medicakolkata_ccu2 | [238.4, 76.5, 721.59 |
| 11 | medicakolkata_ccu1 | [339.600006103515 |
| 12 | medicasiliguri_icu_r | [398.4, 309.1, 665.8 |
| 13 | medicakolkata_ccu2 | [319.6, 122.8, 741, 1 |
| 14 | medicakolkata_ccu3 | [295.5, 137.3, 822.5 |
| 15 | medicakolkata_ccu3 | [443.200012207031 |

Figure 1: Monitor Dataset

**Classification Dataset** The given classification dataset consisted of a combination of images and csv files. There were 4 different sets - each containing 250 images and corresponding 4 csv files with data for the 250 images. The data in the csv files was in the form of bounding box coordinates for each class. Here, class signifies the parameters displayed on the monitor - heart rate, blood pressure, spo2, etc. The bounding box coordinates are given as ( x_center, y_center, x_width, y_width ). As with any real monitor, some values are missing in the original image of the monitor, and thus, the dataset consisted of some null values. These values had to be converted into a string form of float zeroes to make it compatible with the rest of the data.

The data had to be sorted first to ensure correct mapping of images and csv data. It was then reshuffled to allow a fair split of train and test data.

To understand the number of null values in the dataset, we merged the 4 csv files and created one big dataset. From that, we tried to see the trend in null values for each class corresponding to a parameter on the monitor. We also scanned through the 4 files separately to see which dataset had the most amount of null values. In the below image, for the setwise images, each row represents the number of null values in a particular class.

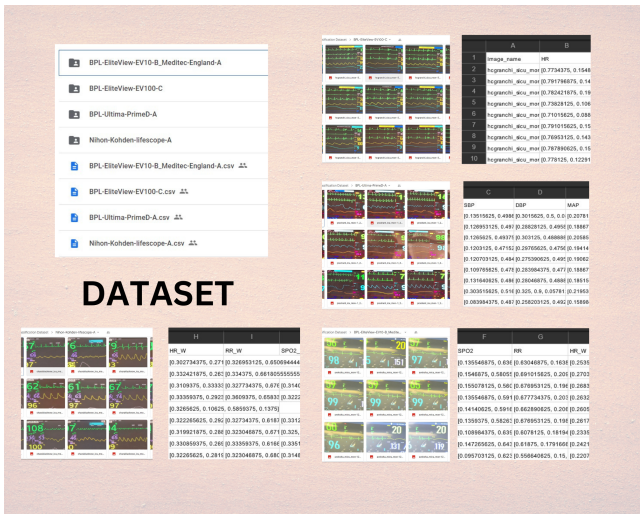


Figure 2: Classification Dataset

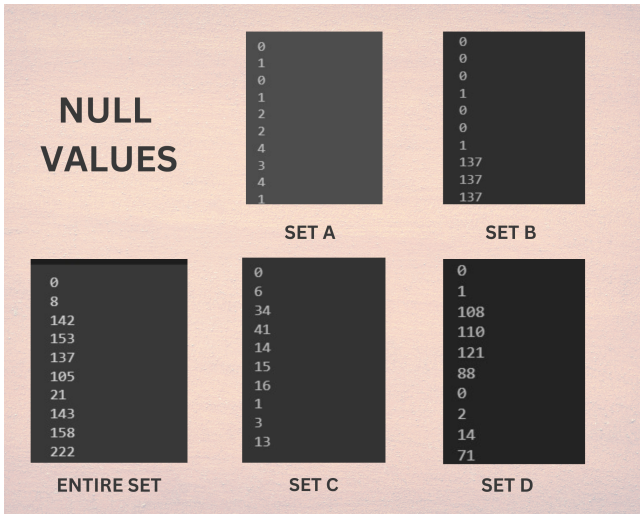


Figure 3: Null Values

**Unlabelled Dataset** The unlabelled dataset comprised of 7000 unlabelled images. It wasn't mapped with any data corresponding to bounding box coordinates for the details of the vitals. This set was primarily used to evaluate the performance of model M2 - the one where we are actually recognizing the values of the vitals. The way the model's performance was evaluated in this case was by checking how our model works if the input image belongs to a different class than the ones we have trained the model on.

## Main Pipeline

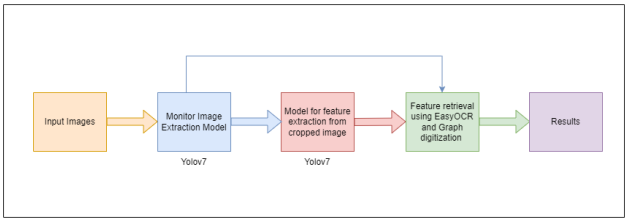


Figure 4: Basic Structure of Pipeline

The complete pipeline designed to extract the human vitals from patient monitoring screens consists of three main stages: Monitor Screen Extraction, Feature Extraction, and Feature Retrieval.

In the Monitor Screen Extraction stage (Model M1), the input image is processed using YOLOv7 to predict the boundary boxes of the monitor screen. The predicted boxes are used to crop the monitor screen from the input image, which is then passed to the next stage.

The Feature Extraction stage (Model M2) uses YOLOv7 to identify individual features within the cropped monitor screen image. The boundary boxes of the identified features are stored in a Pandas dataframe, which is then passed to the next stage.

In the Feature Retrieval stage, the dataframe of boundary boxes and the cropped monitor image are used to crop individual features and predict the numbers present in the images using Easy OCR. The predicted numbers are stored in an output dictionary, and the cropped graph images are processed using masking, thresholding, and the DBSCAN clustering algorithm to digitize and plot the graph.

The pipeline provides a comprehensive solution for the extraction and classification of various human vitals from the patient monitoring screen, including Heart Rate, SpO2, RR, Systolic Blood Pressure, Diabolic Blood Pressure, and MAP.

## Model M1

The Model 1 (Monitor Screen Extraction Model) uses YOLOv7, a popular object detection framework, to predict the boundary boxes of the monitor screen in the input images. The goal of this model is to accurately identify the monitor screen in the images so that it can be cropped and used as input for the next stage in the pipeline.

**Yolov7** YOLOv7 was chosen for this task due to its ability to accurately detect objects in images, even in complex and cluttered environments. This is particularly important for extracting monitor screens from images taken in real-world patient monitoring settings, which may contain other objects or distractions that could interfere with the accuracy of the detection.

The training data for Model 1 consisted of 1800 images with annotated segmentation boundaries for the monitor screens, and a validation set of 200 images was used to evaluate the performance of the model. The default image size used was 640 x 640, and the batch size was 8. The model was trained for 35 epochs using the input format of images and a label (txt file) containing the bounding box coordinates of each class.

**Output** The output format of the Model 1 is a tensor that includes the bounding box coordinates and confidence score for each class in the image. The size of the tensor depends on the number of classes identified in the image.

In conclusion, the use of YOLOv7 in Model 1 was crucial in ensuring accurate detection of monitor screens in the input images. The results of this stage in the pipeline were promising, and the model was able to accurately identify the monitor screens in the majority of the validation images, thus providing a solid foundation for the next stage of the pipeline.

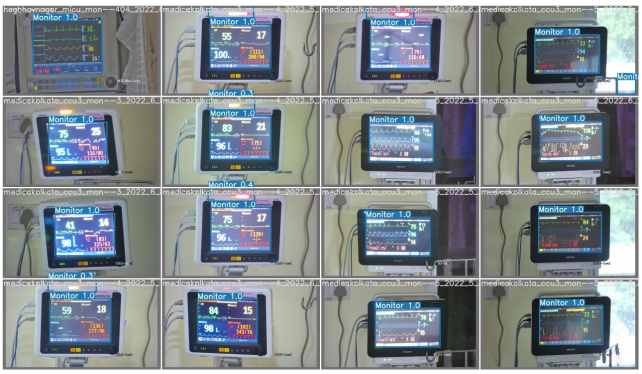

Figure 5: Test Batch for M1

## Model M2

**Classification and Feature Extraction** The M2 model is a classification and feature extraction model, which uses YOLOv7 for object detection. The model takes the cropped monitor image from Model 1 as input and predicts the boundary boxes for individual features in the monitor image. The predicted boundary box coordinates are stored in a Pandas dataframe, which serves as input for the next step of the pipeline.

The model was trained on 800 images and validated on 200 images, with each image belonging to one of four types of monitor screens. The default image size was 640x640 and the batch size used was 8. The model was trained for 45 epochs, with the input format being images and a label file containing the bounding box coordinates of each class in the image. The output format is a tensor containing the bounding box coordinates and confidence score for each class in the image.

**Text Recognition** In the text recognition step, the feature extraction model provides a dataframe of boundary box coordinates and the cropped monitor image. The code crops the image according to the coordinates in the dataframe and uses Easy OCR to extract the readings. The extracted text is cleaned by removing any non-numeric characters and stored in the form of a dictionary with keys as the parameters on monitor and values as the numbers detected.

The use of Easy OCR in this step is important because it provides a simple and accurate way to extract text from an image. This is especially important in this application because the readings on the monitor screen are in the form of text, and accurate extraction of these readings is crucial for obtaining meaningful information about the physiological parameters.

**Digitization of Graph:** The H.R. graph digitization step of the pipeline is designed to extract information from the graph images present on the monitor screen. The code first crops the image based on the boundary box coordinates, followed by thresholding to separate black pixels from the non-black ones, allowing for the creation of a mask to extract the required part of the graph. The centroids from k-means clustering are then used to create a mask and extract the required part of the graph. The final graph is plotted after refining the clustered points with DBSCAN clustering.
The use of thresholding, k-means clustering, and DBSCAN clustering in this step is important because it provides a way to digitize the graph and extract meaningful information from it. The DBSCAN clustering algorithm refines the clustering and separates the data points into distinct groups, eliminating any unwanted noise. The digitized graph can then be plotted to obtain meaningful information about the physiological parameters.

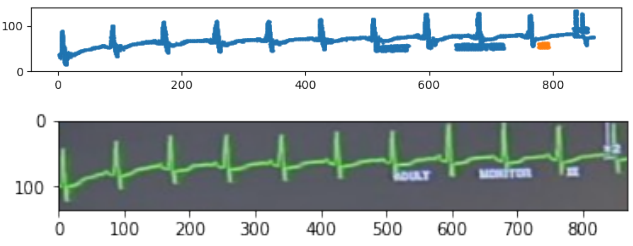


Figure 6: Graph Digitization

In conclusion, the M2 model uses YOLOv7 for object detection, Easy OCR for text extraction, and a combination of thresholding, k-means clustering, and DBSCAN clustering for H.R. graph digitization. These frameworks and techniques were chosen because they provide an efficient and accurate way to extract meaningful information from the monitor images. The model was trained and validated on a dataset of 1000 monitor images, with the resulting predictions providing accurate information about the physiological parameters.

Figure 7: Test Batch for M2

## Results

**Inference Time Analysis** We ran our model in Google Colab Notebooks. We tested our pipeline on public clusters of Google Colab notebooks. For test inputs, the average inference time of our pipeline is around 3.5 seconds. This time includes the time for all the stages of our pipeline, i.e., image preprocessing, and feature extraction, predictions and its processing.

**Model wise results** The accuracy of the pipeline is directly related to the accuracy scores of both the models. The following images depict the results achieved by models 1 and 2, respectively. We have depicted each model's confusion matrix, F1, and PR curve.

For model M1 we have achieved an F1 score approaching one at a confidence of 0.912

For model M2, from the PR curve, we have achieved a decent precision and recall (over 0.8) in extracting the information regarding all the vital metrics. The F1 curve indicates that all the classes have an F1 score of 0.93 at a confidence of 0.702. .

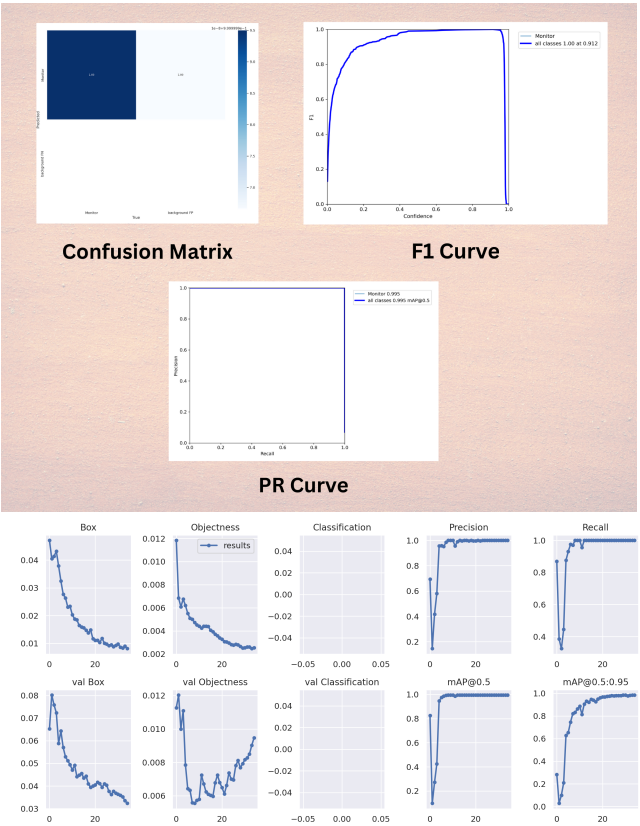


Figure 8: Results for Model M1

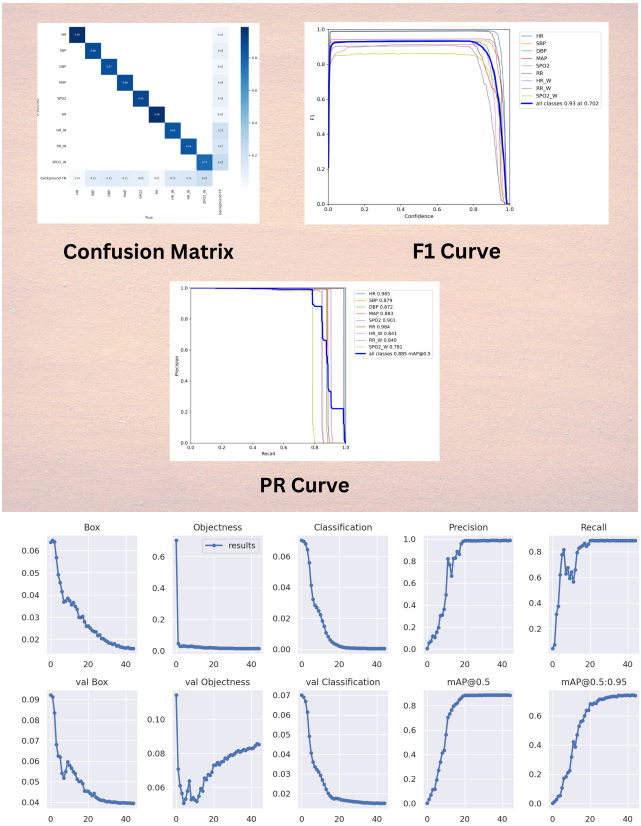


Figure 9: Results for Model M2