INTER IIT TECH: MID PREP

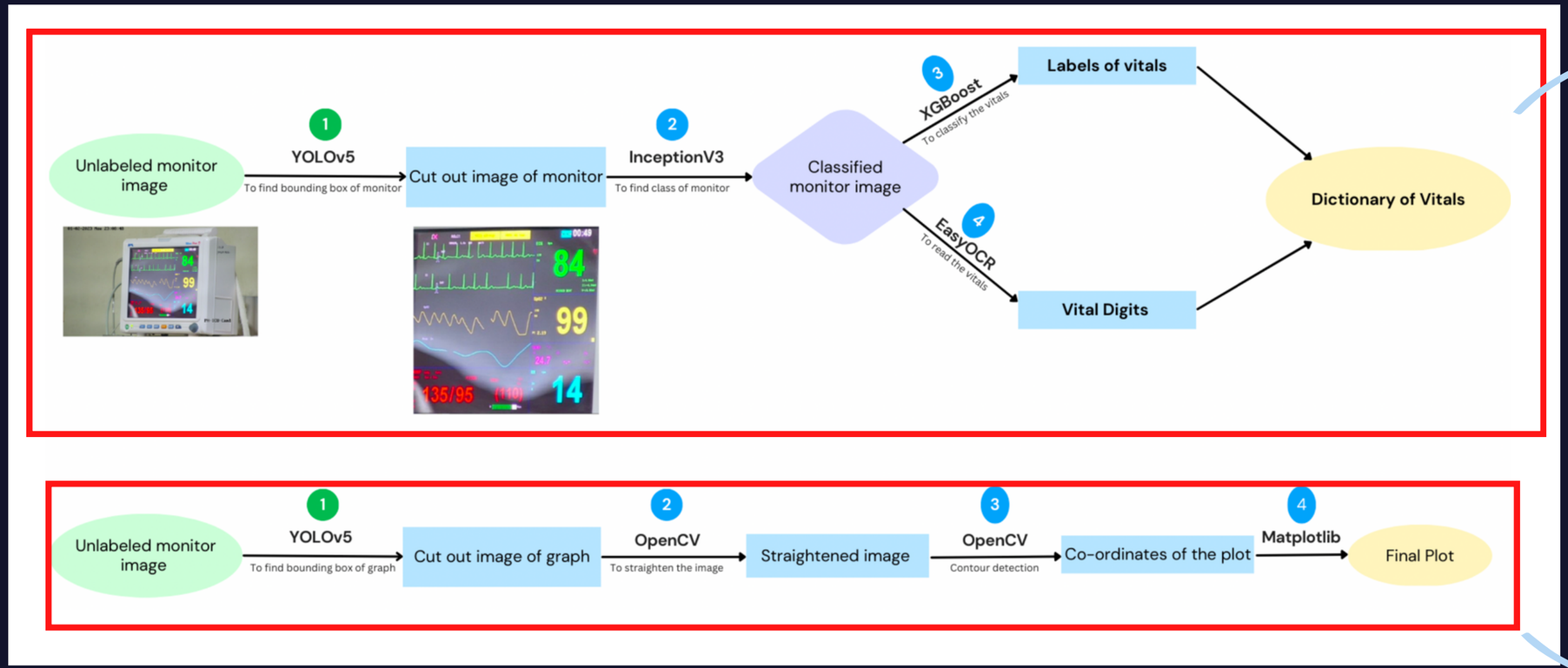
CLOUD PHYSICIAN

OVERVIEW OF OUR APPROACH

- FOR THE GIVEN PROBLEM STATEMENT WE PROPOSE AND BUILD A TRANSFER LEARNING BASED, MULTI LAYERED PIPELINE ARCHITECTURE.
- OUR PROPOSED MODEL BEING AN INTERPRETABLE PIPELINE HELPS TO LOCATE ERRORS EFFECTIVELY AND SCALE UP EASILY.
- WE GIVE AN IMAGE CONTAINING AN ECG MONITOR AND AT THE END OF THE PIPELINE WE ARE GIVEN A PYTHON DICTIONARY WITH THE IDENTIFIED LABELS, ALSO A MATPLOTLIB PLOT DISPLAYING THE DIGITIZED HEART RATE CAN BE OBTAINED.

PIPELINE ARCHITECTURE

Our model mimics the process of human vision to identify and read the vitals.



MONITOR SEGMENTER

- To identify the monitor from an image, we used YOLO v5 (You Only Look Once) which is a state of the art model for quick object detection.
- The model will give us the x-center, y-center, height, and width of the monitor detected, then we select the one with the maximum confidence score and crop out the monitor and pass it to later stages.

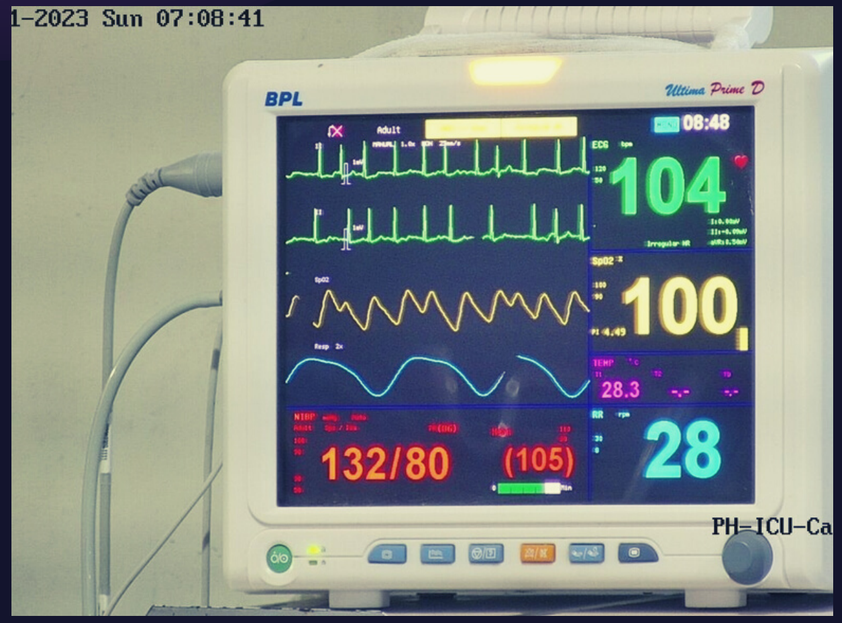
- To train the model we trained the yolo v5 model on the 1800 of the 2000 image samples provided and tested the model on the other 200 images, the training was done with 25 epochs and 8 as batch size.

Precision	Recall	mAP50	mAP50-95
0.99	1	0.995	0.99

MONITOR SEGMENTER

THIS IS THE MONITOR SEGMENTATION WHICH IS DONE YOLO V5 MODEL

MONITOR SEGMENTER



MONITOR CLASSIFIER

- Once we found the bounding box of the monitor, we classy the monitor into four classes by using K-means clustering.
- For this we used a deep learning model, Inception V3 which is based on CNNs. This is documented as the image_feature function in the final pipeline which gives us the label of the monitor as output.
- Initially we tried a Resnet-18 based transfer learning model for Monitor Classification, but we realised we should be taking Monitor Layout as the key feature instead of the explicit Monitor Type and hence preferred an unsupervised learning algorithm.
- This approach resulted in 100% monitor classification over the labelled data.

DIGIT SEGMENTER

- Again to detect the digits, we used YOLO v5.
- This model returned the x-center, y-center, height, and width of the detected digits. To prevent false positives, only the top 6 digits were selected based on their confidence scores and were then cropped for further processing.
- To make the model machine agnostic, we trained it on all four classification datasets provided and treated all the vitals as the same class. In our initial attempts, we tried to get yolo to both segment and also classify the labels, however the results were discouraging
- The following results were obtained when trained for 30 epochs with 10 batch size.

DIGIT SEGMENTER

metrics/recall	metrics/mAP_0.5	metrics/mAP_0.5:0.95	val/box_loss
0.99627	0.9942	0.80342	0.015804

VITAL CLASSIFICATION

- The digit's location and size information obtained from the previous step is fed into an XGBoost classifier to predict its vital type.
- The model was trained on a total of around 250x6 samples per class, with a grid search performed to optimise the hyperparameters.
- For this, an XGBoost model was specifically trained for each of the four monitor classes. The position and size of the digit's bounding box was used as input features for the model to make predict its class.

Monitor Class	Accuracy
BPL-Ultima-PrimeD-A-classification	100%
BPL-EliteView-EV10-B_Meditec-England-A	98.69%
BPL-EliteView-EV100-C	97.0%
Nihon-Kohden-lifescope	99%

OCR-DIGIT RECOGNITION

- To detect text from images of various vitals, we used EasyOCR, which employs deep learning algorithms for accurate and efficient text recognition.
- EasyOCR, Pyterreact, and Calamari OCR were all tried, and EasyOCR showed better accuracy as it allowed us to filter out the output and only consider numeric value.
- Also before being passed to OCR, we are increasing the sharpness and using auto brightness to remove glare in the image.

87

HR - DIGITIZATION

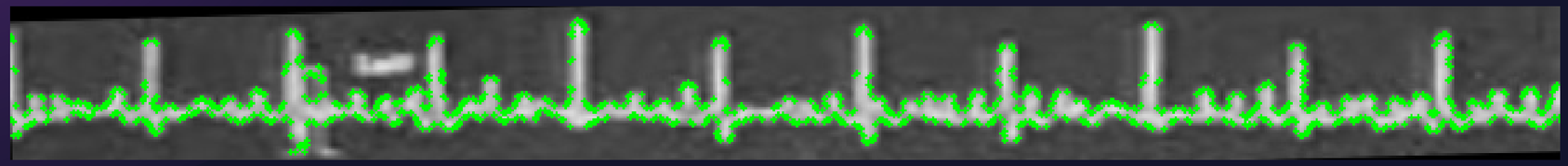
- We trained a separate YOLO v5 model on the whole Classification Dataset to get the cropped HR graph which was then used for contour detection using openCV library.

Original Graph

- We rotated the cropped image by an angle to align all the peaks and hence get the same y-coordinate for all the peaks.

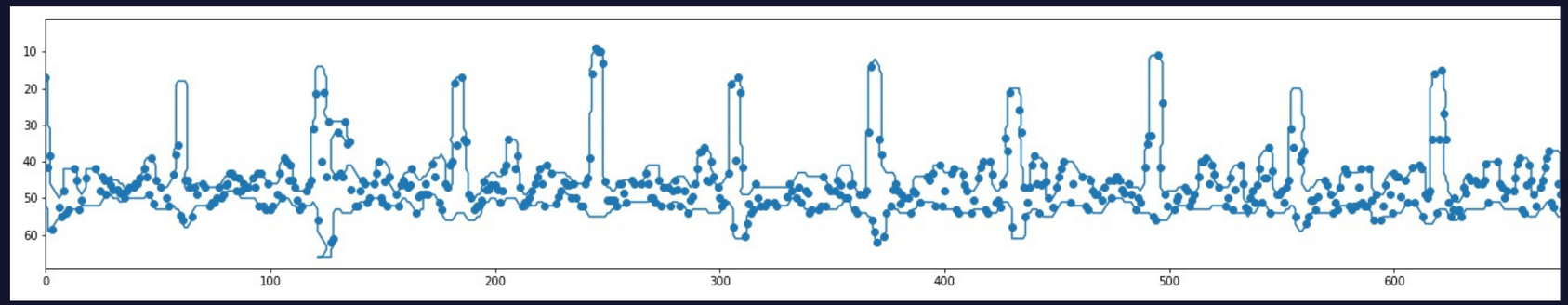
Straightened Graph

- Contour detects the border, i.e. both the inner and outer borders of the curve.



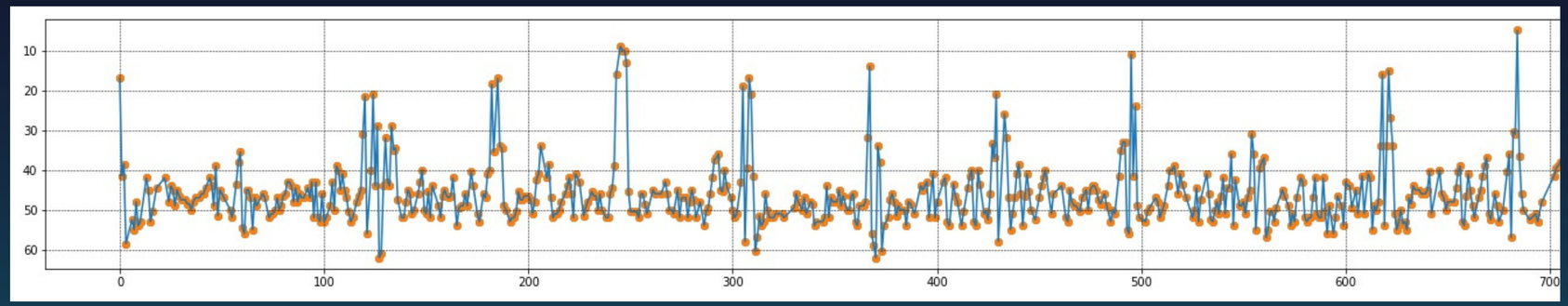
Contour Detection

- This gives us multiple y-coordinates for a single x-coordinate.



Contour Plot

- To resolve it, we take the mean of all y-coordinates



Digitization Plot

THANK YOU!