

HR : 52

SBP : 138

DBP : 79

52

138

79

SpO2 : 100

100

(105)

MAP : 105

22

RR : 22

Team 45

Cloudphysician's The Vital Extraction Challenge






Objectives

What an ideal solution should comprise

- 1. Accurate Vital Extraction**
Considering the crucial nature of the task, the extracted information should be as accurate as possible
- 2. Fast Inference**
The inference should be fast, considering usage in real-time application
- 3. Robustness**
The algorithm should be robust to multiple layout, so better generality

Stages



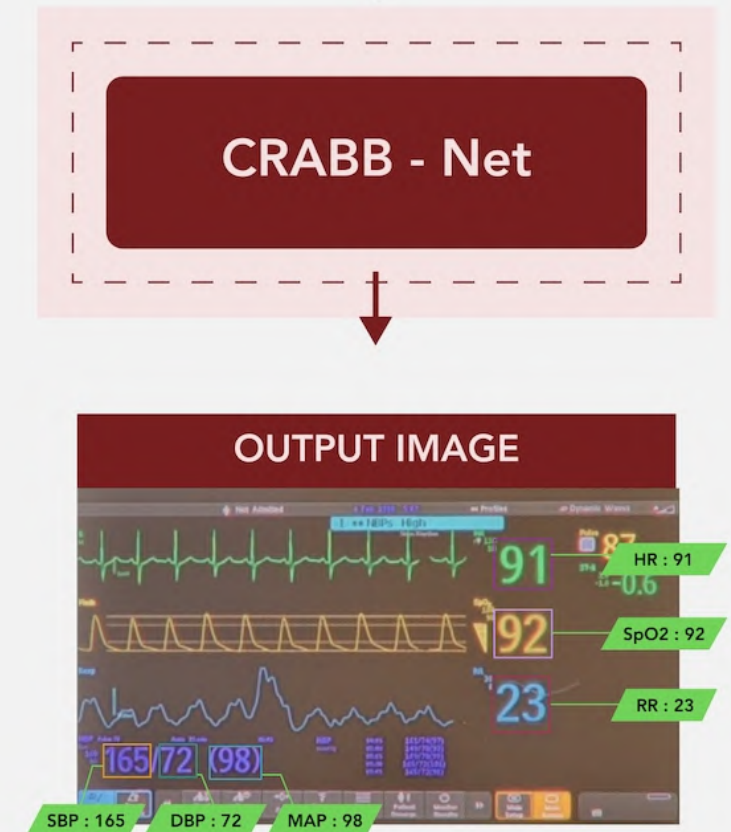
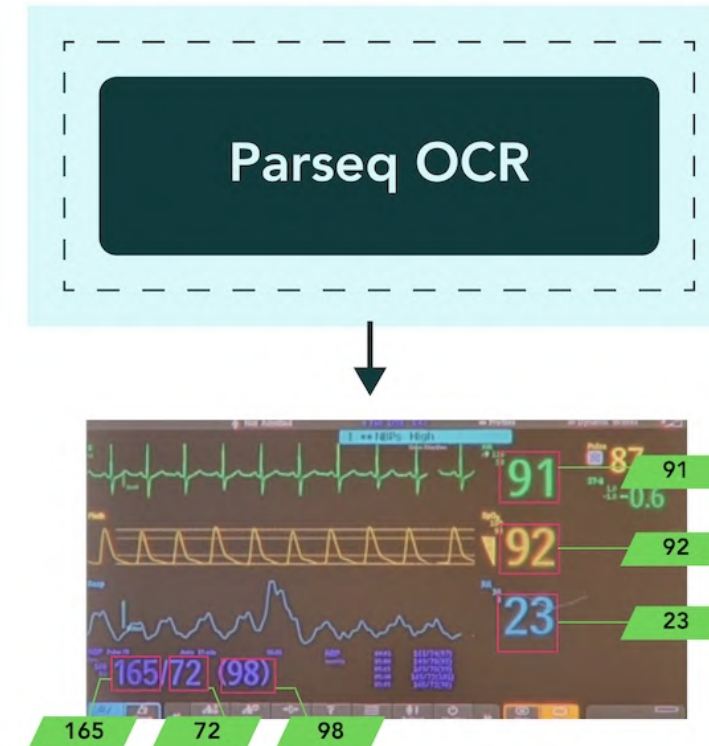
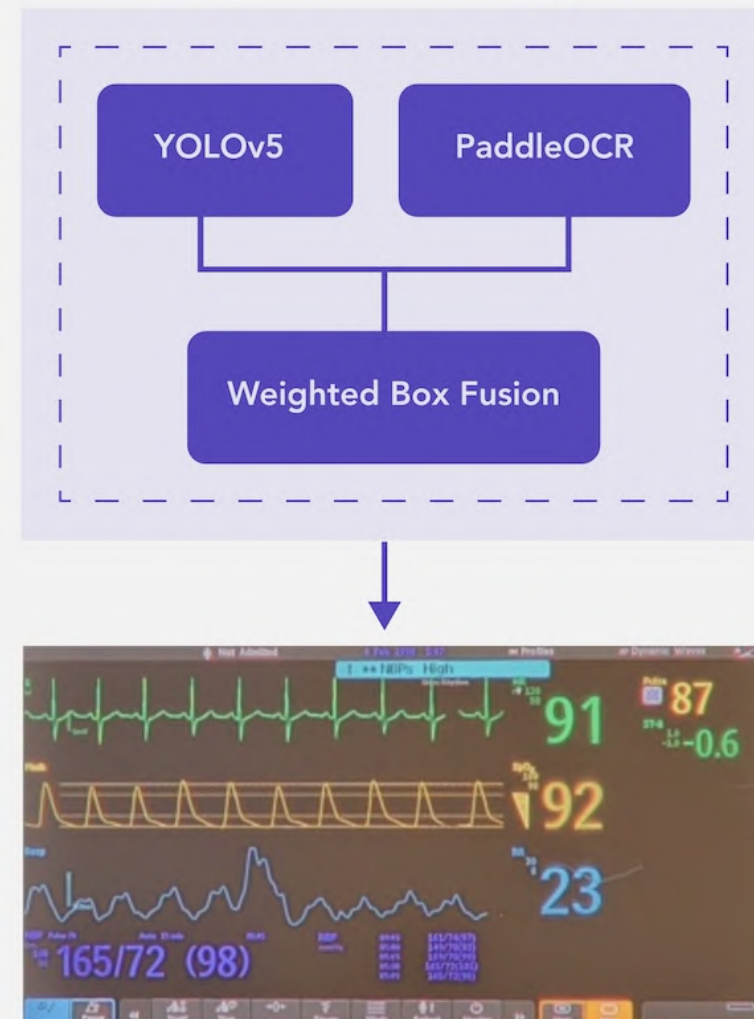
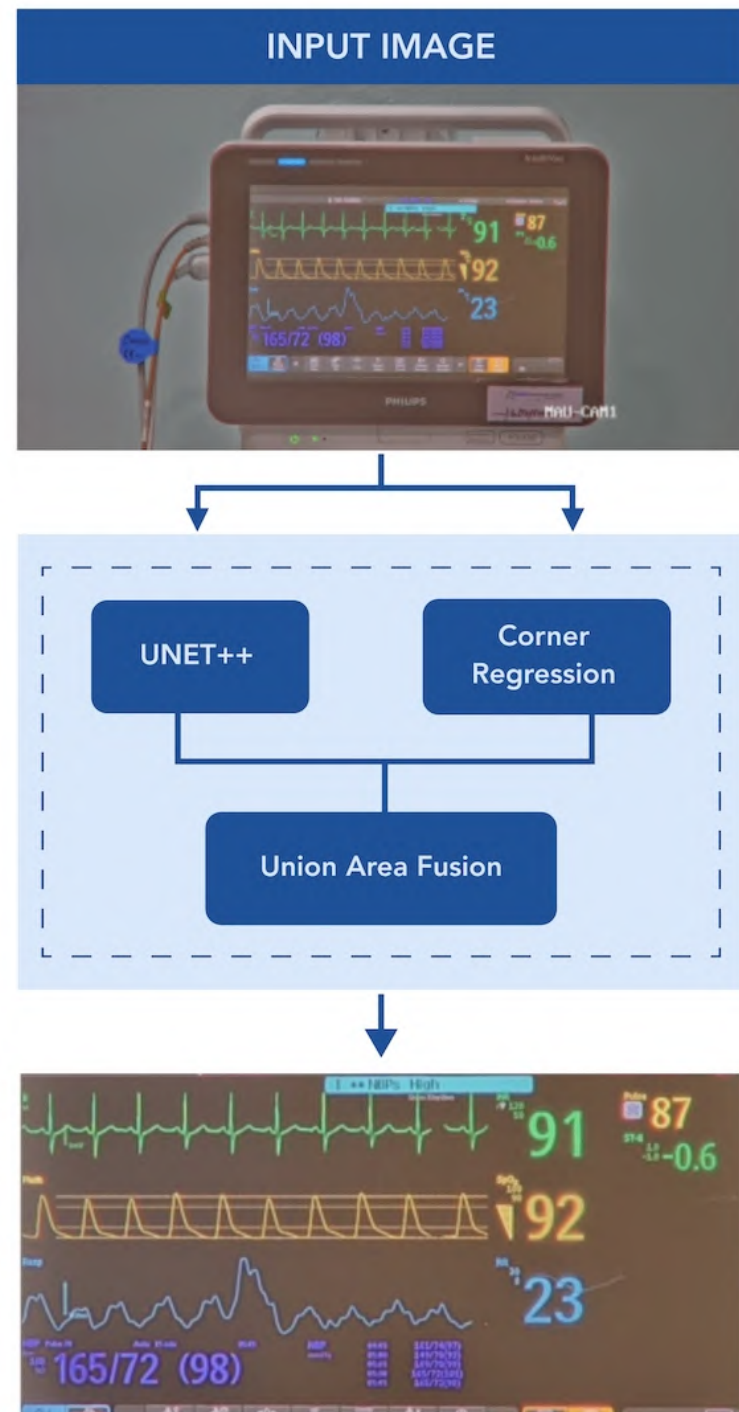
Screen
Extraction

Digit
Localization

Digit
Recognition

Vital
Mapping

Pipeline



Screen Extraction

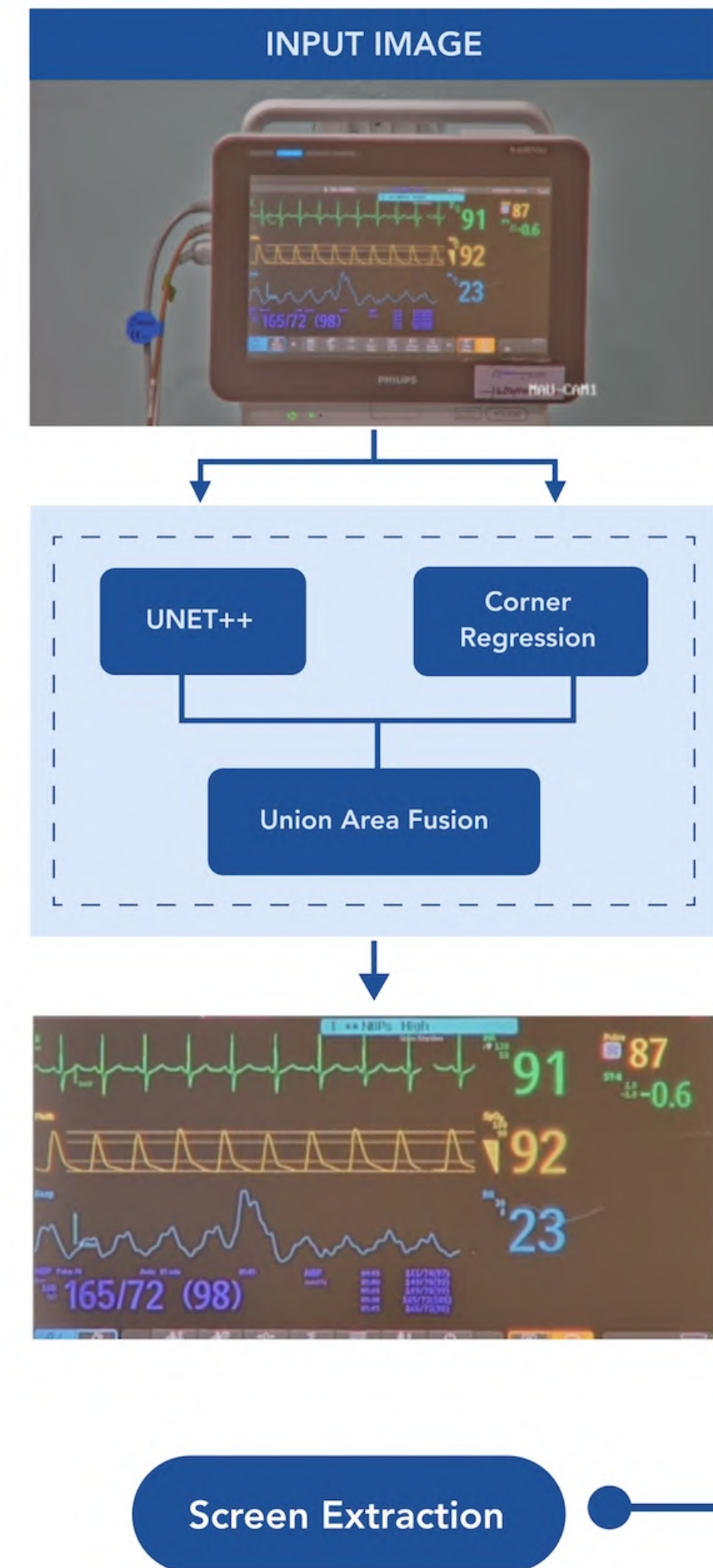
Digit Localization

Digit Recognition

Vital Mapping

Extracting Screens from Images

Screen Extraction



Corner Regression

It uses a CNN encoder to extract the four corners of the screen as a regression task

Custom Augmentations such as Horizontal Flip and RandomCrop, along with RandomBrightness Contrast for Robust Performance

Projection Transformations, using homography matrix to convert predicted bounding boxes to rectangular images

Inference Time: 0.32 s
Mean IoU: 0.873



UNET++ Semantic Segmentation

Semantic segmentation to produce mask of the screen from the input image

Combo Loss function combining Dice Loss and BCE Loss for increasing both IoU and confidence of predicted mask

Custom Post Processing Techniques to improve mask, and extract accurate bounding boxes from prediction

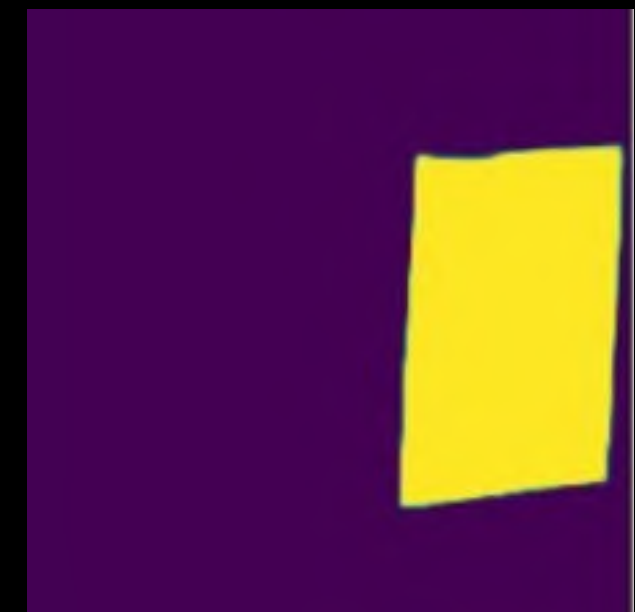
Input Image



True Mask

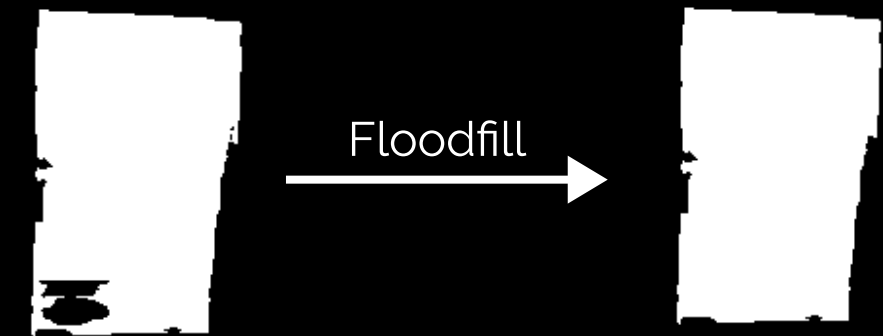


Predicted Mask

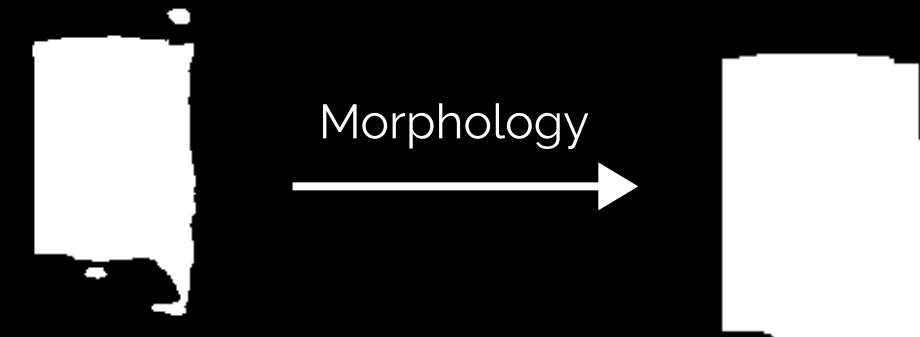


Post Processing for UNET++

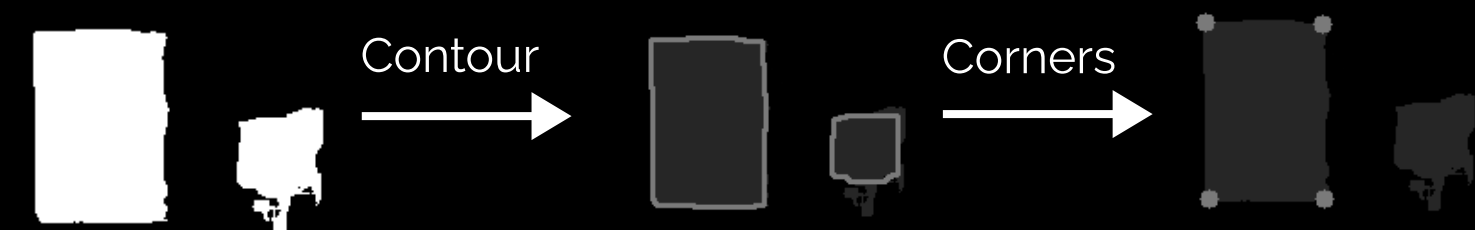
Floodfilling for filling open holes in predicted mask



Morphological Functions to smoothen the predicted mask



Contour Detection for extracting contour from mask, and area maximization for final corner detection



Inference Time: 4.71 s

Mean IoU: 0.896

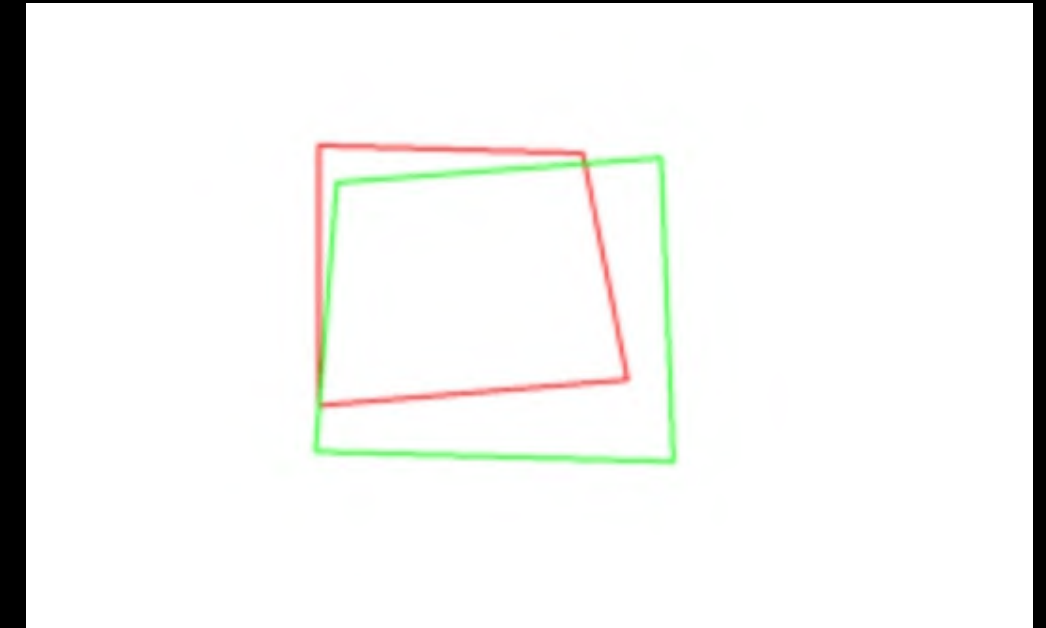
Area Union Fusion Ensembling

Combines Prediction of Corner Regression and UNET++ to ensemble for final prediction

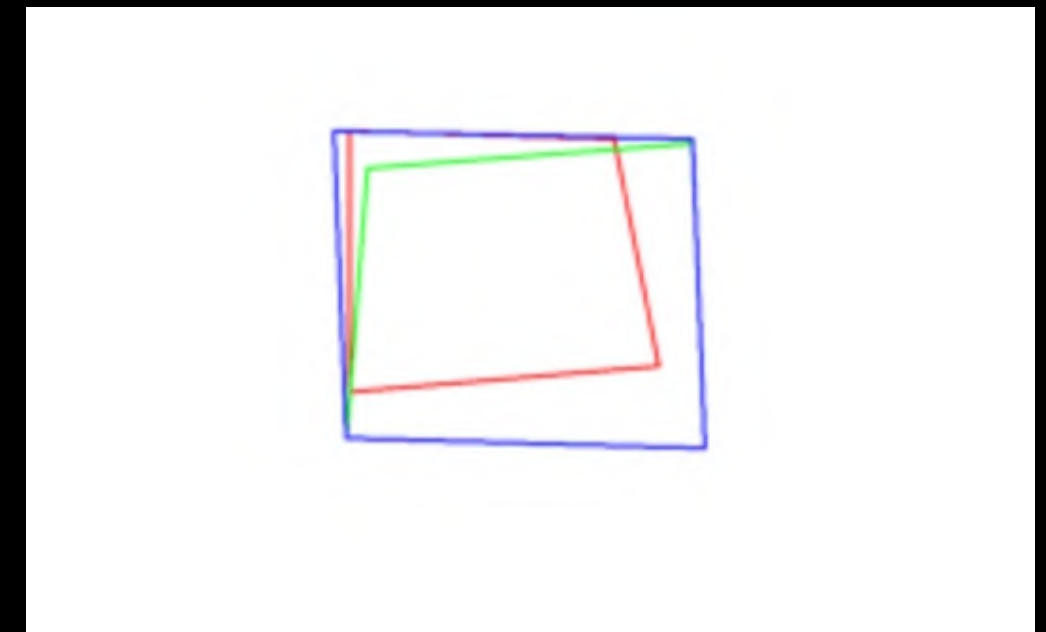
Finds Largest Polygon Covering Both Predicted Polygons, ensuring maximum area coverage

Inference Time: 5.24 s
Mean IoU: 0.910

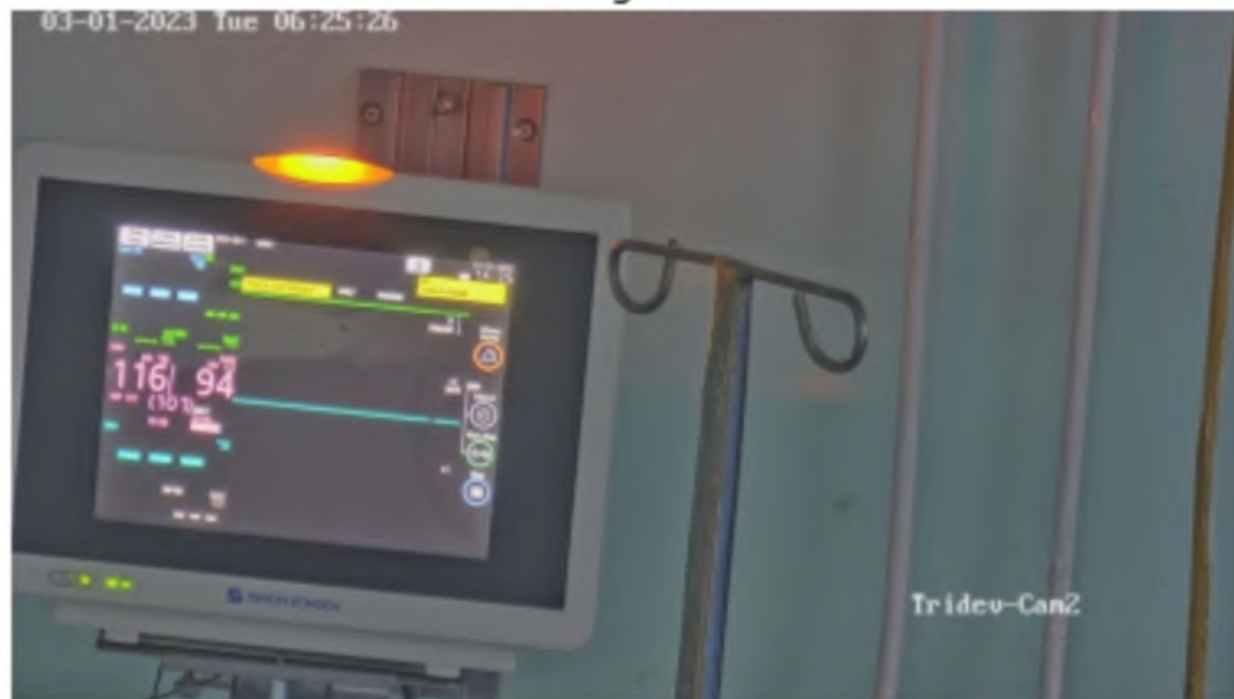
Before Fusion



After Fusion



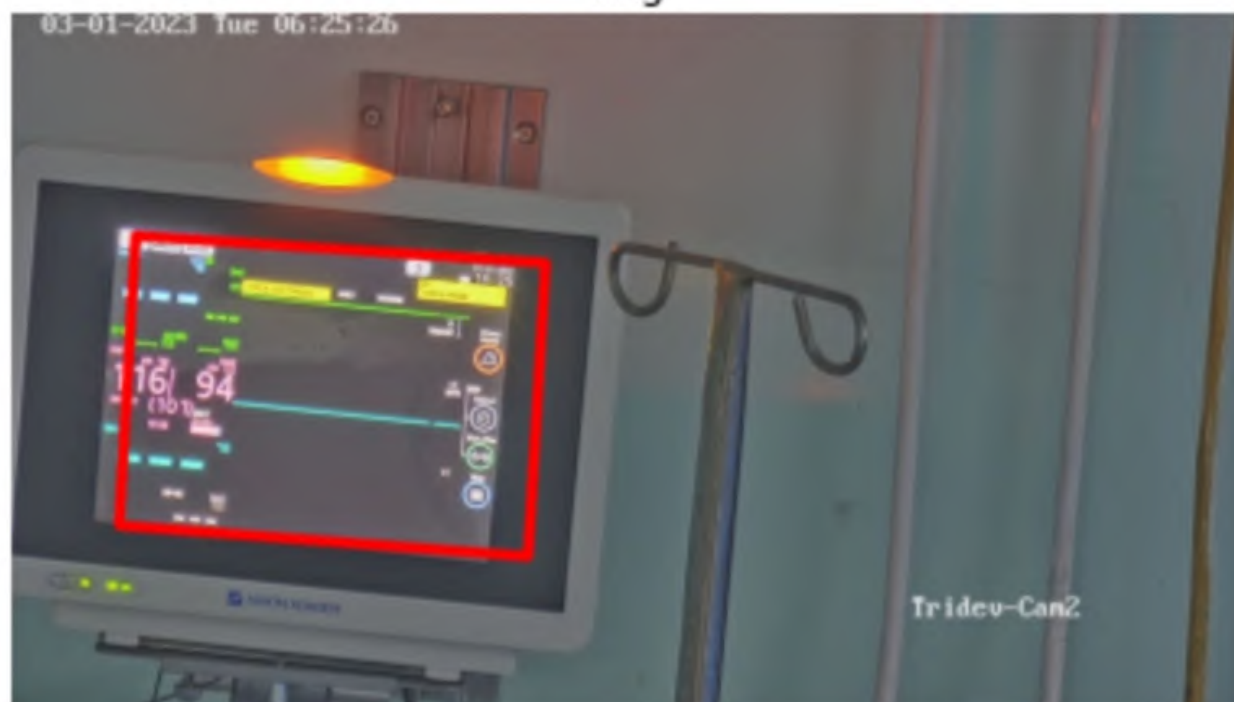
Original



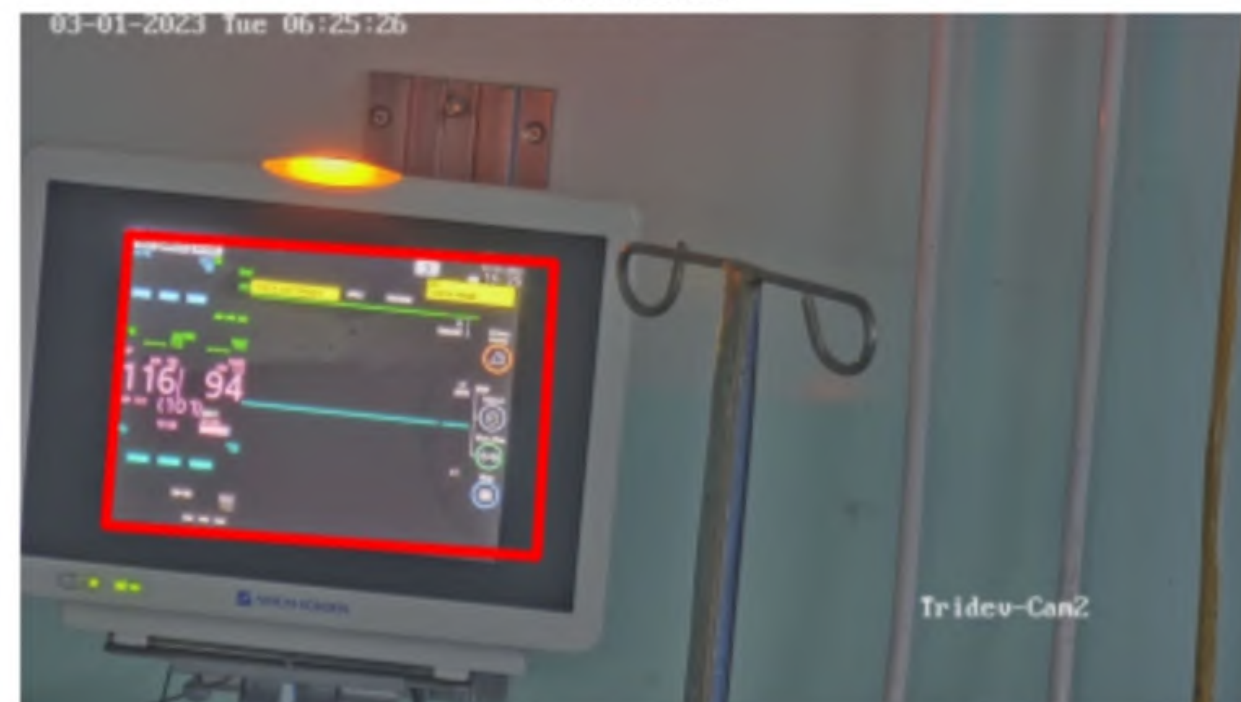
Unet++



Reg

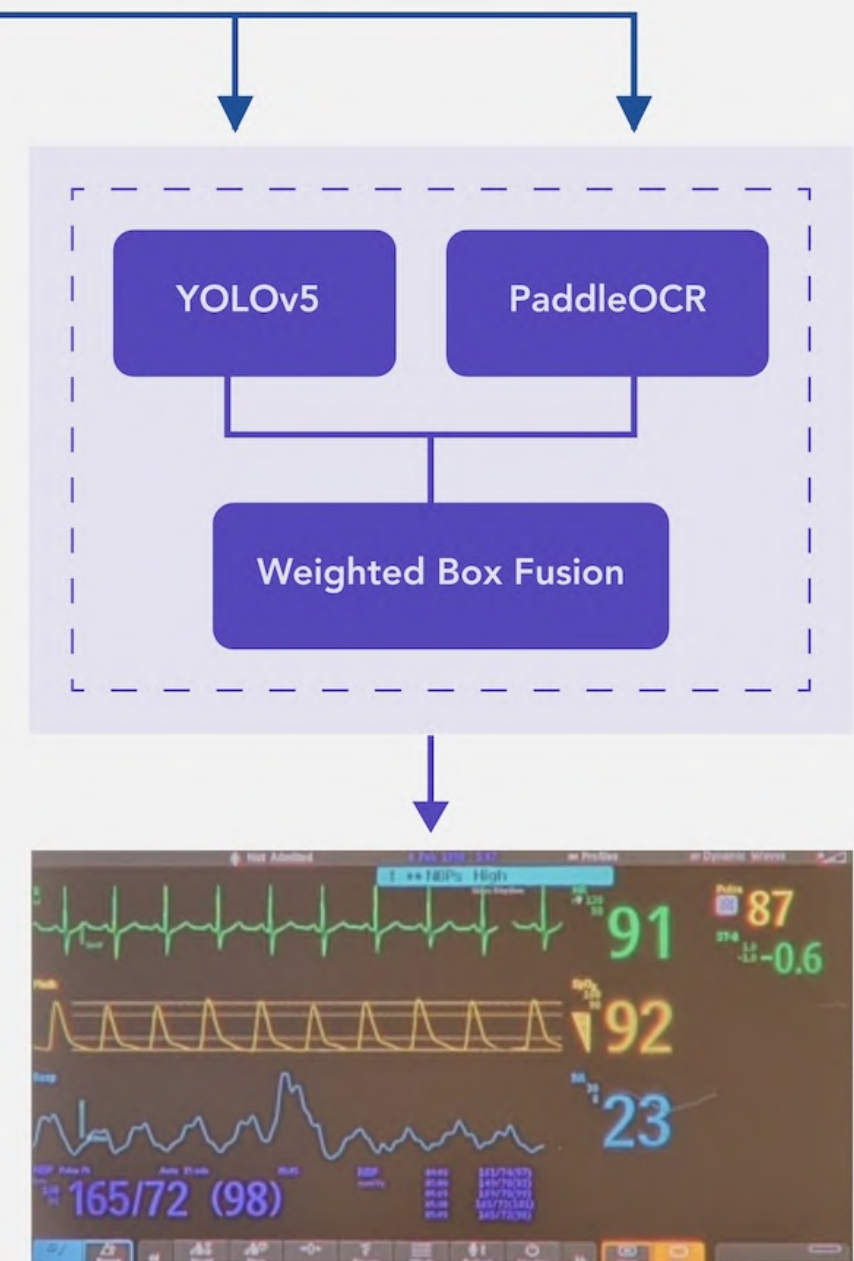


Ensemble



Localizing numbers on screen

Digit Localization



Digit Localization

YOLO

Using the extracted screens from the monitors, we proceed to extract numbers from it. We used YOLOV5m due to its smaller size and less inference time (700 ms)

The Model was trained on the provided 1k annotations and 500 annotated images selected using low confidence thresholds.

To Improve the detections from YOLO we ensemble the results with PaddleOCR text detections.

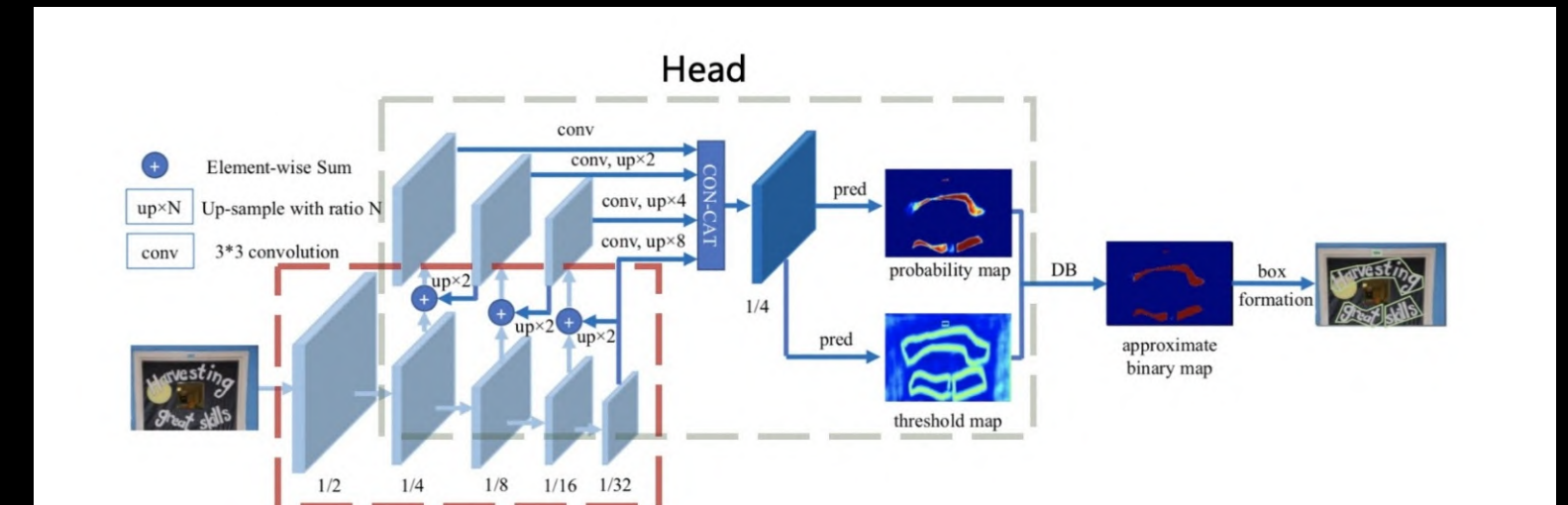


PADDLE OCR for detection

PaddleOCR uses a Differential Binarization network as its feature extraction and detection algorithm.

The Pretrained knowledge of Paddle OCR supplements the YOLO model that was trained on the ICU data. It also has comparable inference time to YOLO (300 ms)

PaddleOCR also detects text boxes which were filtered out using our recognition algorithm outputs.

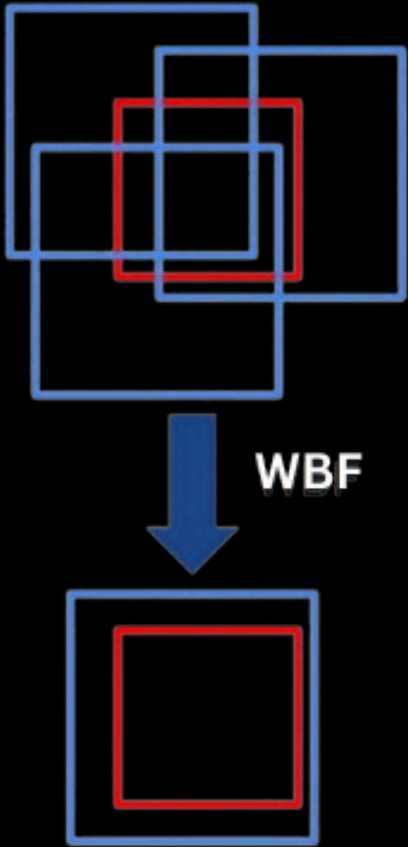


Weighted Box Fusion

Weighted Boxes Fusion is a novel method for combining predictions of object detection models. It utilizes confidence scores of all proposed bounding boxes, as weights, and constructs weighted bounding box.

This ensures that no vital numbers are being missed and produces tightly bounded boxes around the numbers without any noise for the OCR to read.

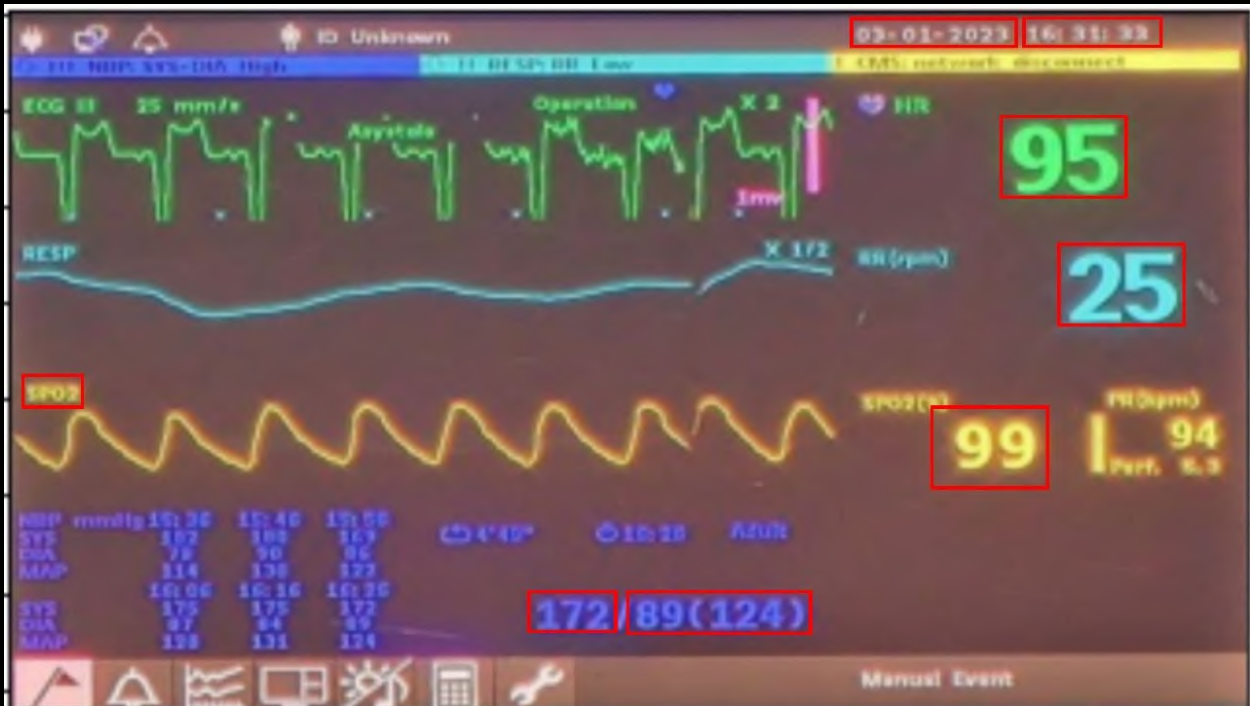
Finally, a filtering algorithm is designed to remove boxes containing noise, using recognition output. After this step, we are only left with boxes containing vitals. The inference time of this stage is 1.6 s



Model	AP (at 0.3)
PaddleOCR	0.67
YOLO	0.79
WBF ensemble	0.84

METRICS

Prediction of PaddleOCR



Prediction of YOLO

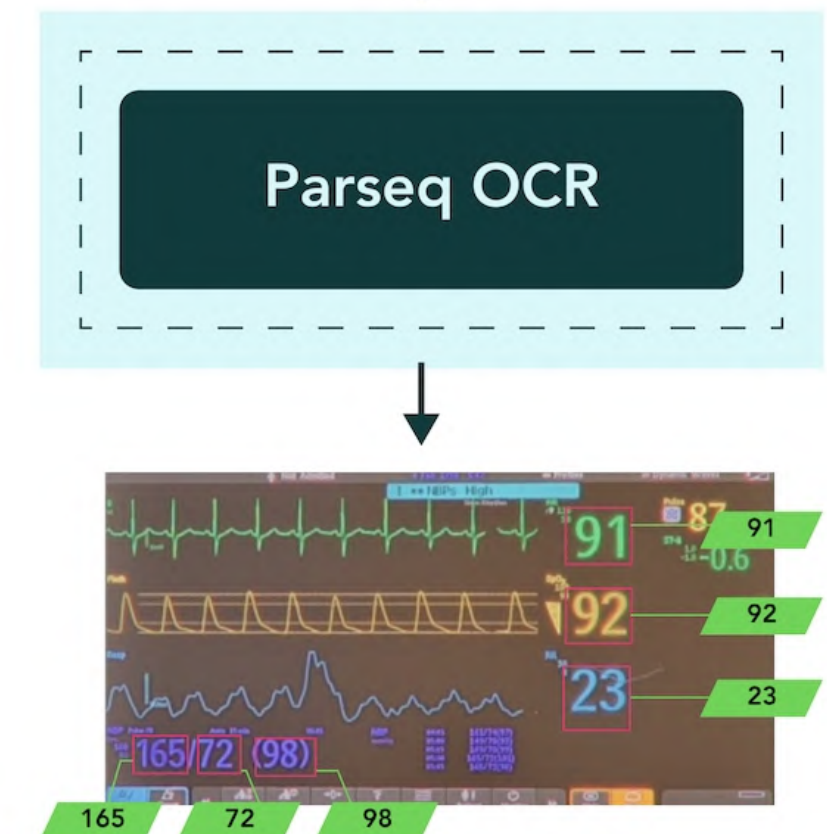


Weighted Box Fusion + Filtering



Recognizing numbers on screen

Digit Recognition



Digit Recognition

PARSeq

We used **Parseq**, a **Scene Text Recognition** model to read data from the bounding boxes.

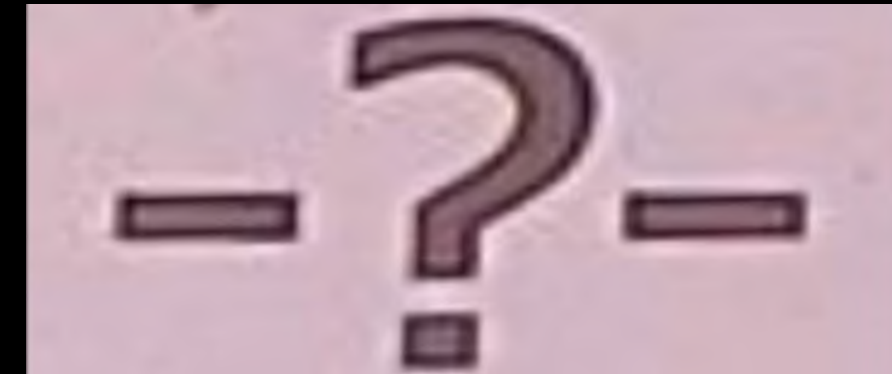
Parseq proved to be better than other models while extracting information from obscure images/ images having misaligned text from the dataset.

Using batch inference we improved the latency by 23%.

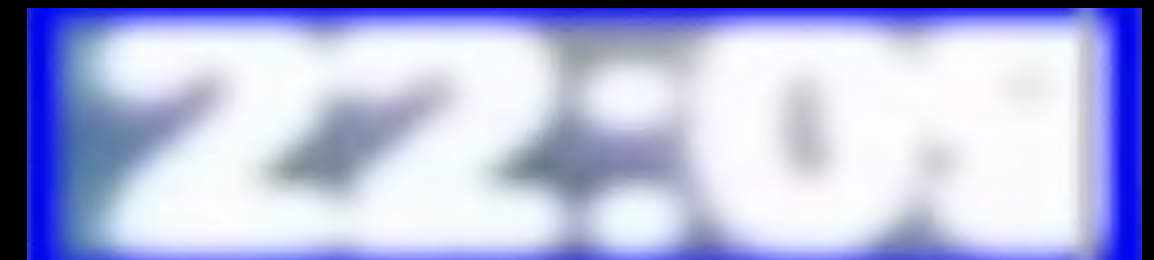
Latency : 880 ms

Accuracy : 98.4%

(calculated on 1032 tightly cropped YOLO predictions)



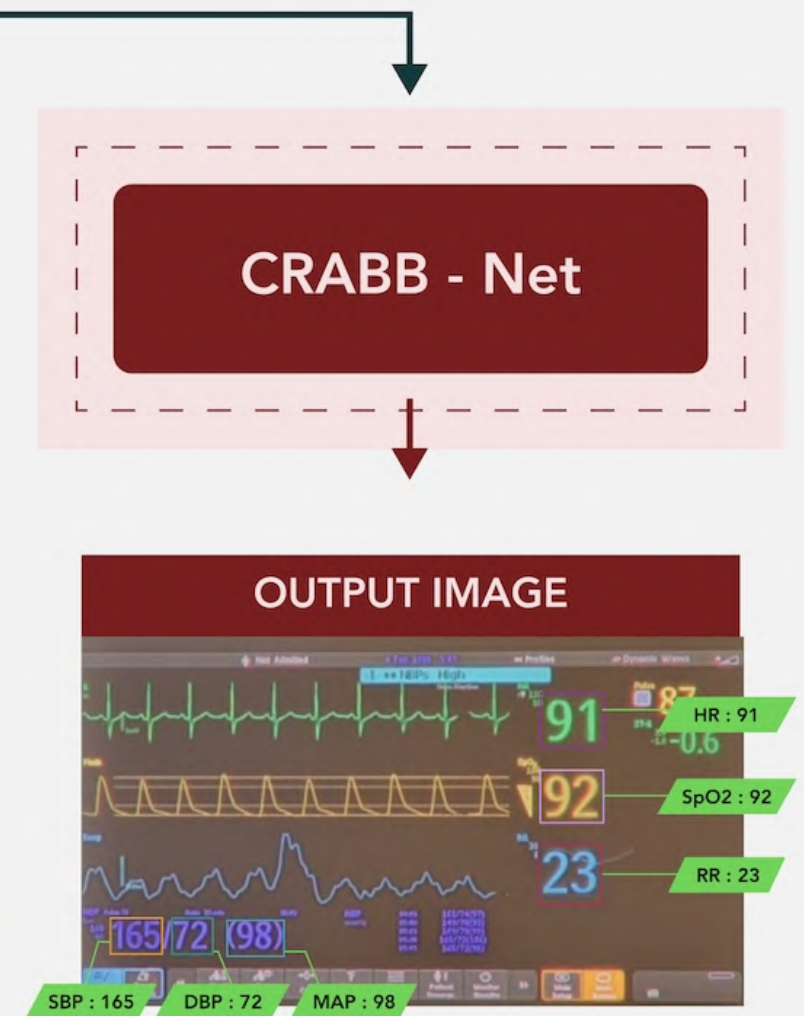
Output : '-?-'



Output : '22:09'

Mapping numbers to vitals

Vital Mapping



Vital Mapping

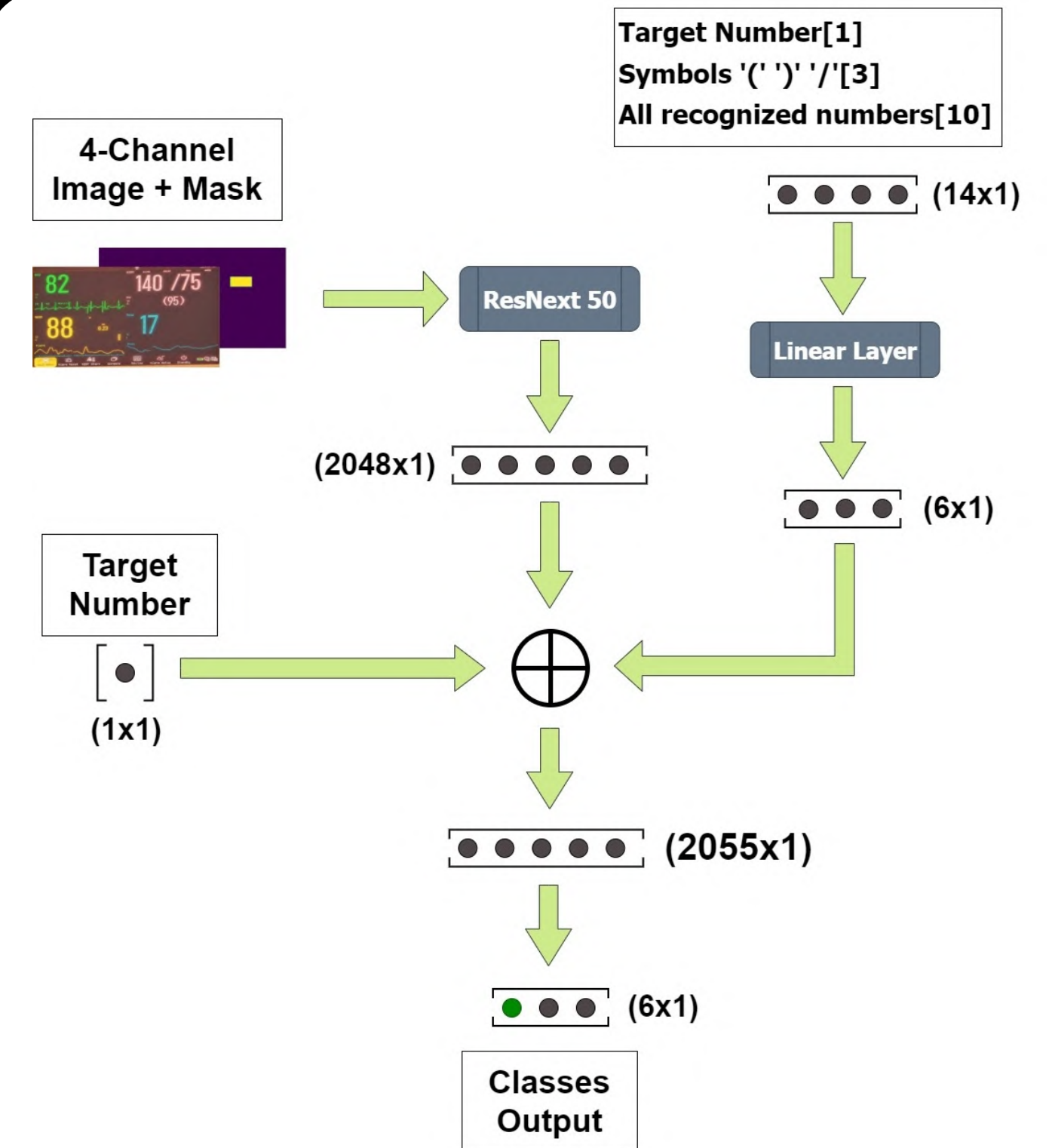
CRABNet

To map each recognised output to the vital we design the architecture

Custom Recognition Assisted Bounding Box classification Network (CRABBNet)

For this task we input the image, bounding boxes and all numbers, and associated text data recognised in the screen to classify the vitals.

We use the confidence scores of all numbers corresponding to vital classes and pass them to our custom logits-decoder to map them.



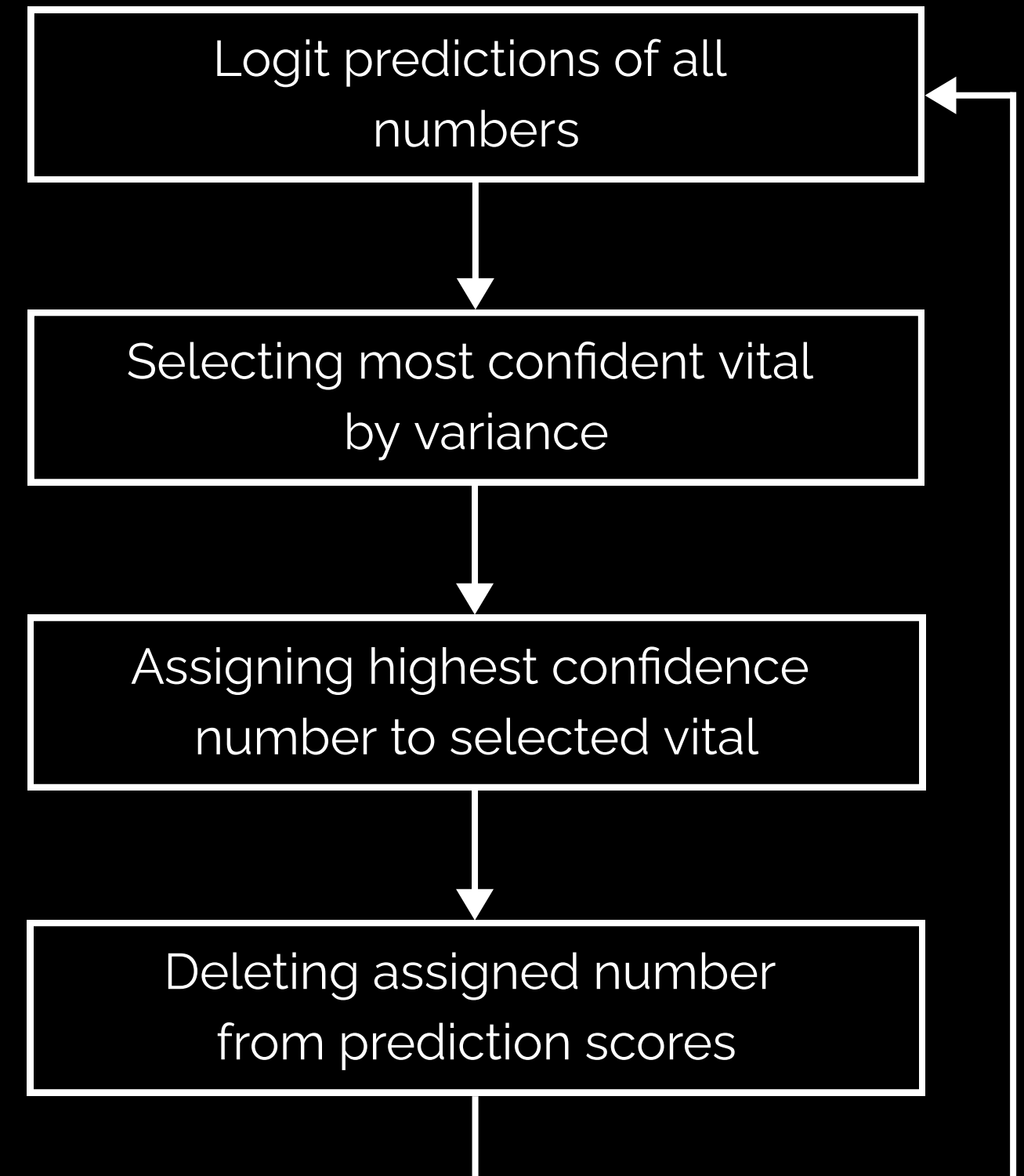
Custom Logit-Decoding for Multi-label mapping

To constrain one output for each prediction, we design a custom logit decoding algorithm.

Using the confidence scores for each class, we first select most confident class using variance and assign the box with the class.

Recurring for all the classes we finally extract all boxes with the vitals.

Using the algorithm, we can also correctly decode the presence of a vital as the best box is chosen for each vital. The inference time of CRABNet is 1 s.

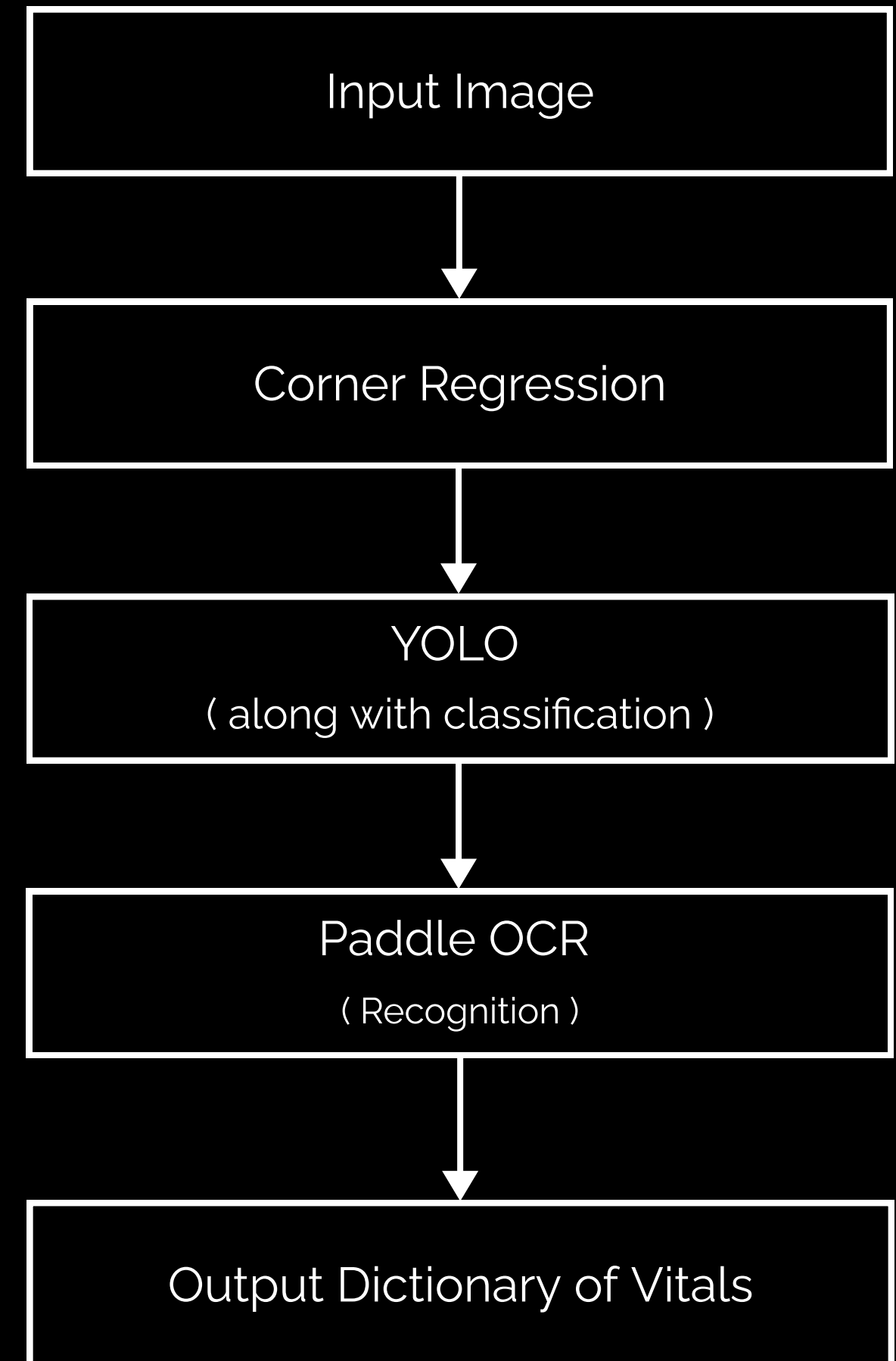


Fast Mode

The Fast Mode ensures the entire pipeline runs under 2 seconds on a CPU.

In Fast Mode, we are using just Corner Regression for Screen Extraction. Extracted Screens are passed to YOLO model, for detection and recognition of vitals directly.

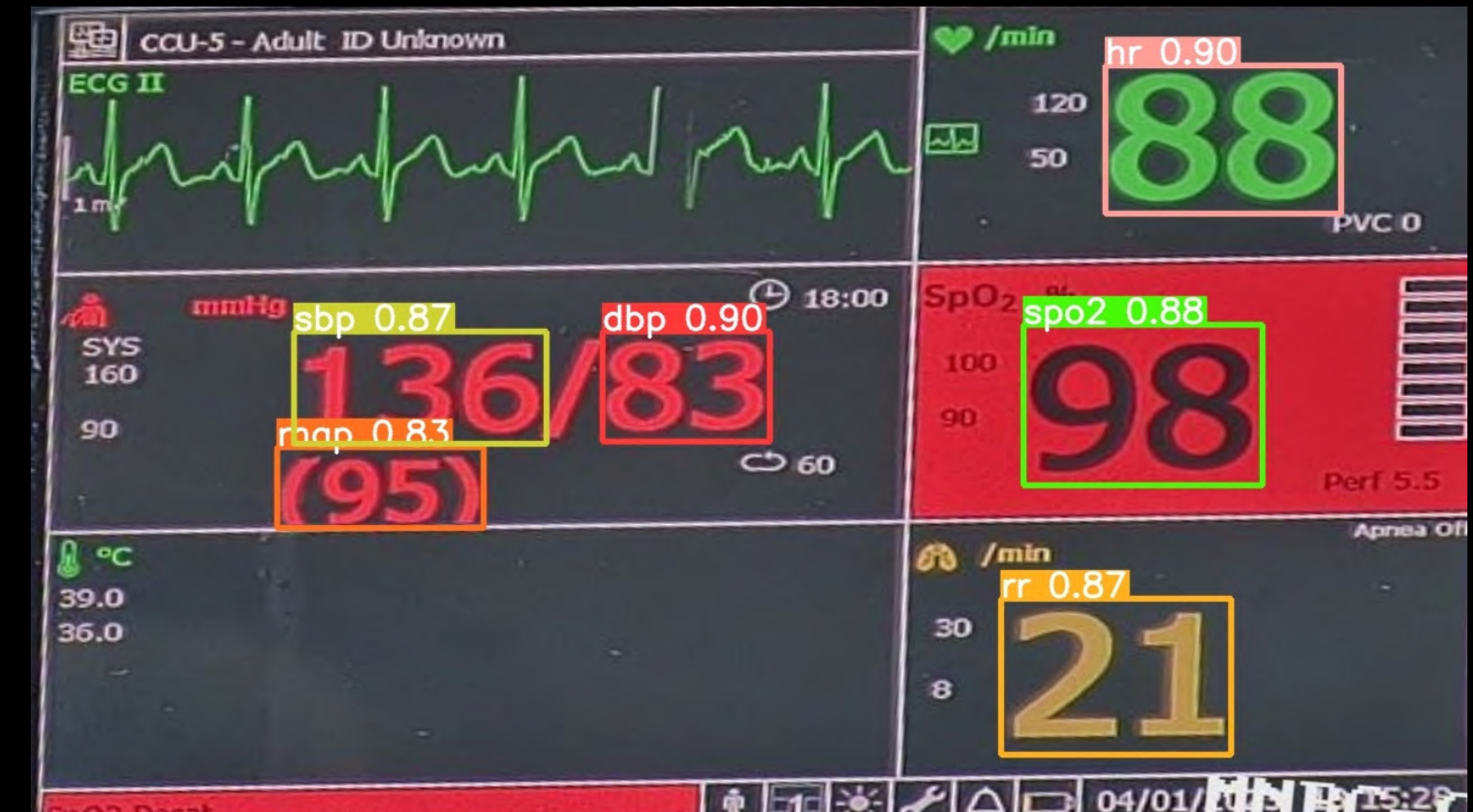
The recognized vital images are then passed to PaddleOCR, for reading the vitals. The inference time of this fast mode is **1.57 s**



YOLO (end-to-end Vital Extraction and Classification)

For the fast mode of our pipeline, we use yolo for both extracting the vitals present in the screen and labelling them (with an inference time of 700ms).

Using YOLO for both the tasks improved the inference time of our pipeline by a great extent, but was not able to perform well on the layouts it was not trained on.



ACCURATE MODE

- CORNER REGRESSION AND UNET++ ENSEMBLING FOR SCREEN EXTRACTION
- YoloV5 + PaddleOCR FOR DETECTION WBF WITH FILTERING ALGORITHM
- PARSeq OCR FOR DIGIT RECOGNITION
- CRABB-NET FOR VITAL MAPPING

ACCURACY : 89.7%

TIME : 8 s

FAST MODE

- CORNER REGRESSION FOR SCREEN EXTRACTION
- YOLOv5 FOR DETECTION AND VITAL MAPPING
- PaddleOCR FOR DIGIT RECOGNITION

ACCURACY : 74.1 %

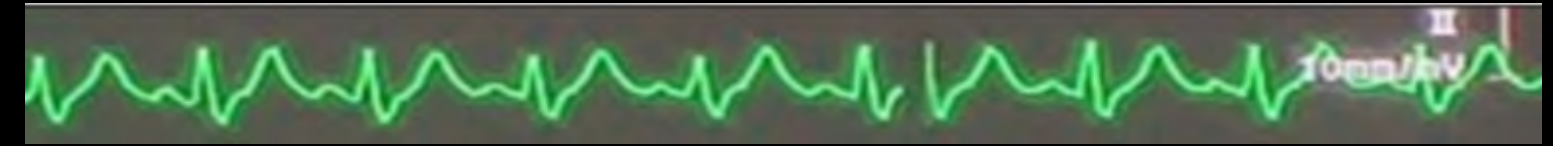
TIME : 1.57 s

Graph Digitization

Yolov5 is used for extracting the HR Graph from the screen. It is trained on the given classification dataset containing 1000 images.

After getting the coordinates of the graph, the input image is cropped and then preprocessed using grey-scaling, thresholding, contour mapping and contour extraction.

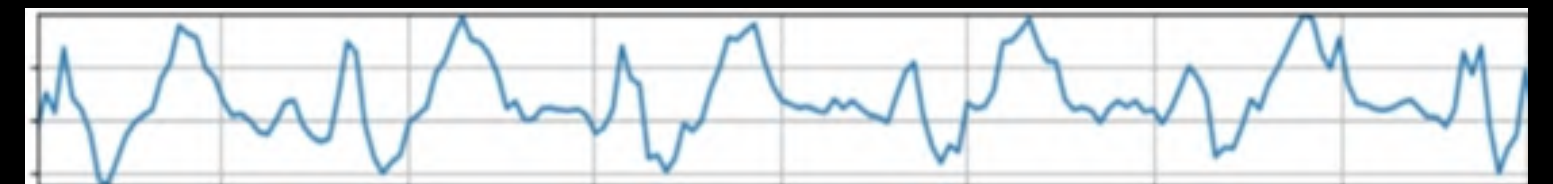
Then the extracted Contour is approximated to get the data points, which are then digitized.



Cropped Image



Extracted Contour



Digitized Graph

Novelties

1. Designing Screen extraction task as a regression problem to predict the coordinates of the boxes using an end-to-end model.
2. Using confidence scores of YOLO predictions to extract hard images for Manual annotations to train Object detection and Vital classification models.
3. Filtering PaddleOCR (pretrained) outputs using recognition algorithm and using Weighted Box Fusion to ensemble the detections.
4. Designed custom architecture, CRABBnet to incorporate recognition predictions and bounding box mask into CNN encoder features.

Novelties

5. Created custom augmentation functions for Corner regression training.

6. Designed logit decoding algorithm to assign one-to-one mapping between recognised numbers and the vitals.

Future Works

Future Works

Custom Number and Symbol Recognition Model based on Individual digit Classification, reducing 1000 classes (possible 3 digit numbers) to 33 classes and symbols. This can be trained on synthetically generated data.

A custom vital classification model to localise numbers & map all numbers to classes simultaneously. For this we propose using a 3 channel to 7 channel UNet which takes image as input and gives a segmentation mask output for each class.

Fine-tuning PaddleOCR on the ICU data for better number detections.

Using image extrapolation techniques on countour graph predictions to predict a better graph.

References

[UNet++ : A Nested U-Net Architecture for Medical Image Segmentation](#)

[PARSeq : Scene Text Recognition with Permuted Autoregressive Sequence Models](#)

[Weighted boxes fusion: Ensembling boxes from different object detection models](#)

[YOLO v5](#)



Thank You