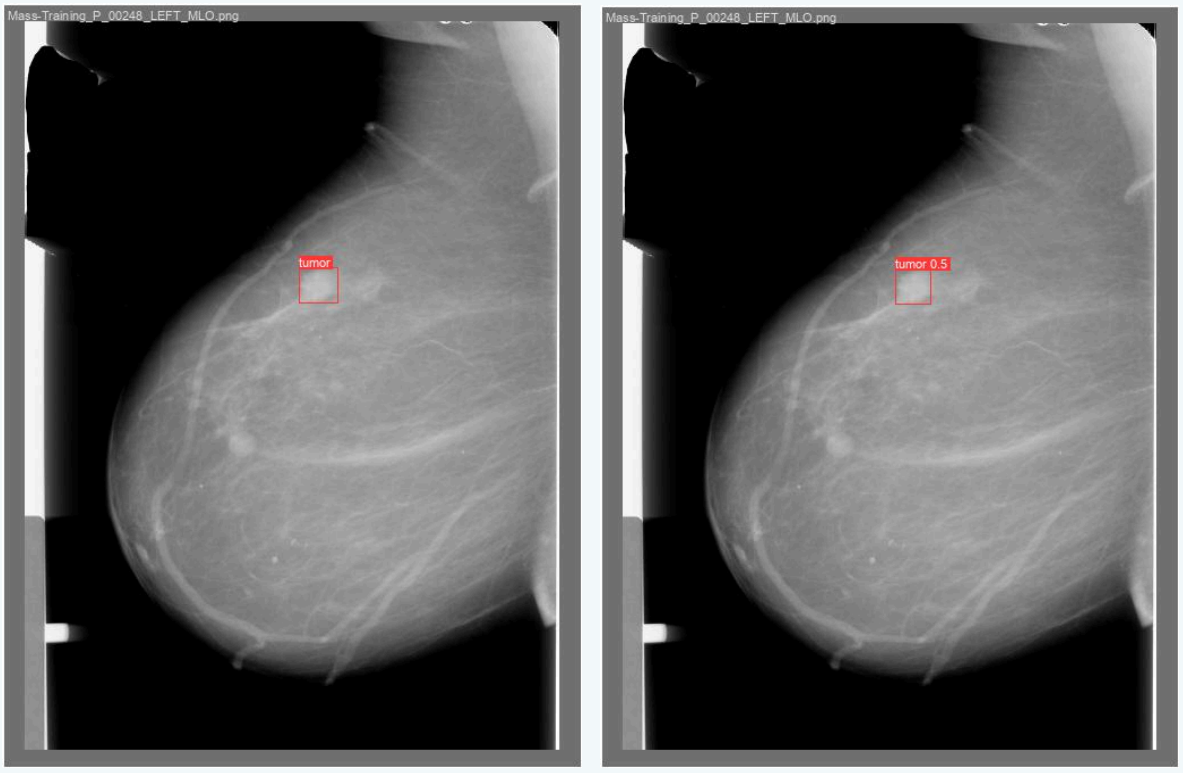
COL780- Assignment 4

Supreeti Kumari 2020CS10396
Eshan Jain 2020CS50424

1) Yolo Based Model:

Data Visualisations:

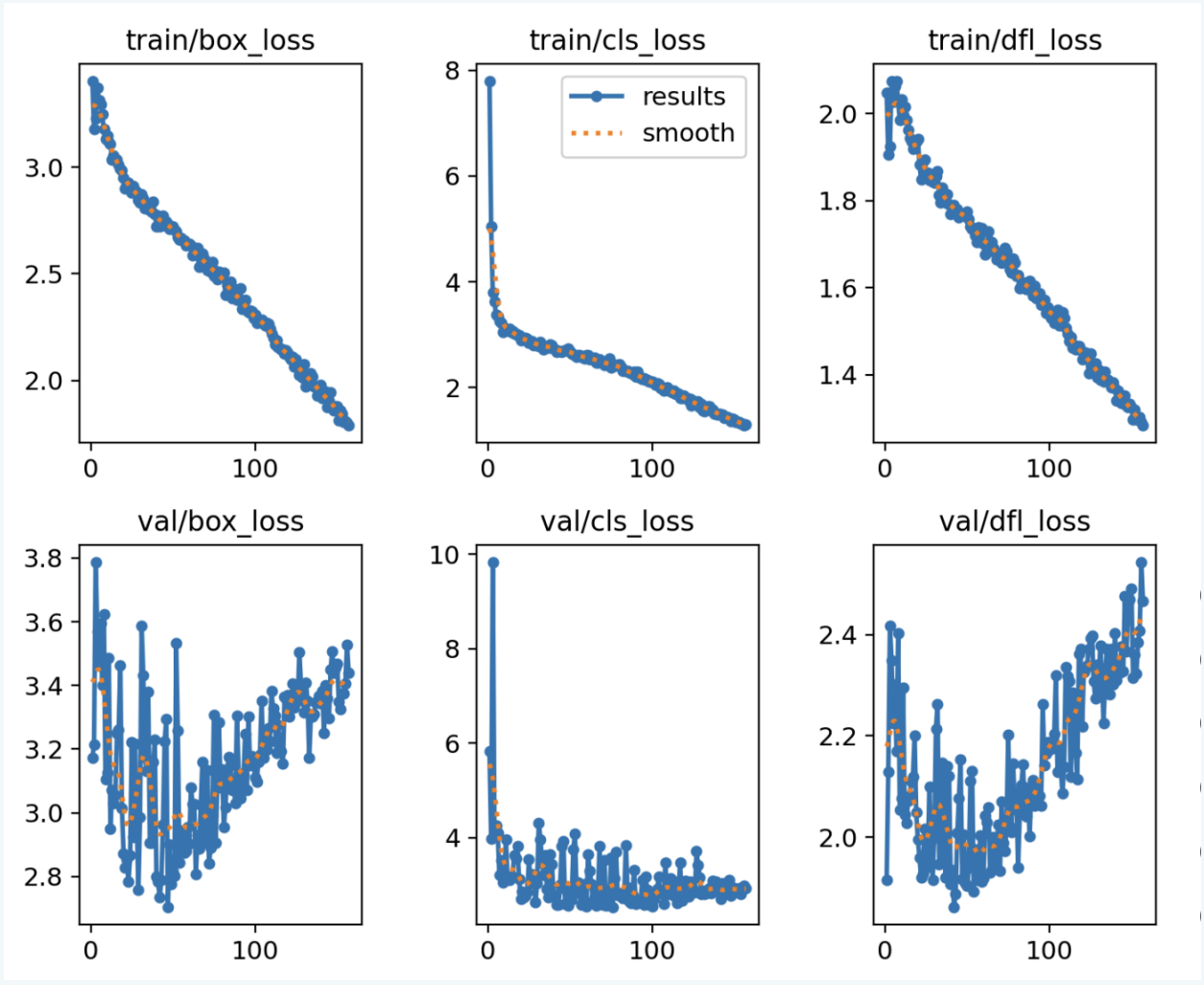
Image Name: Mass-Training_P_00248_LEFT_MLO.png (in yolo/val folder)

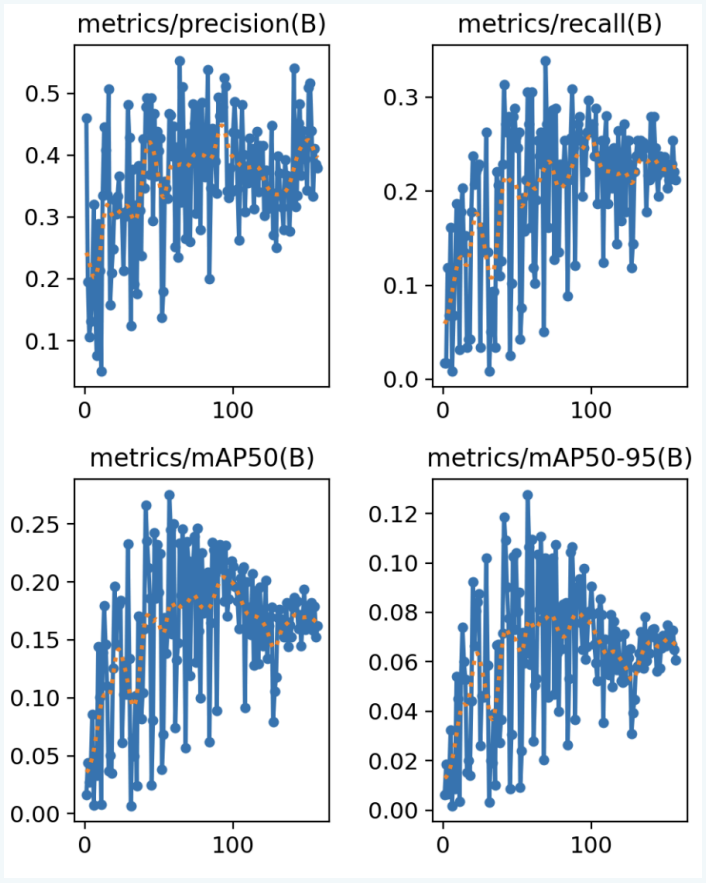


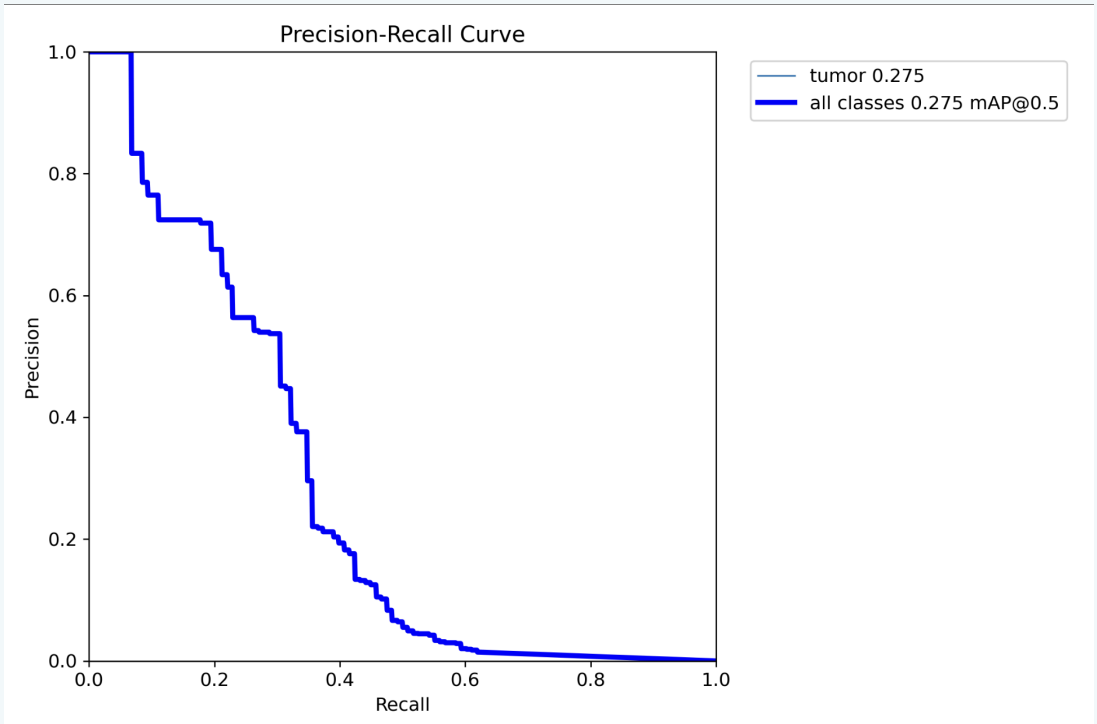
(On Left: Label Bounding Box, On Right: Predicted Bounding Box)

Plots:

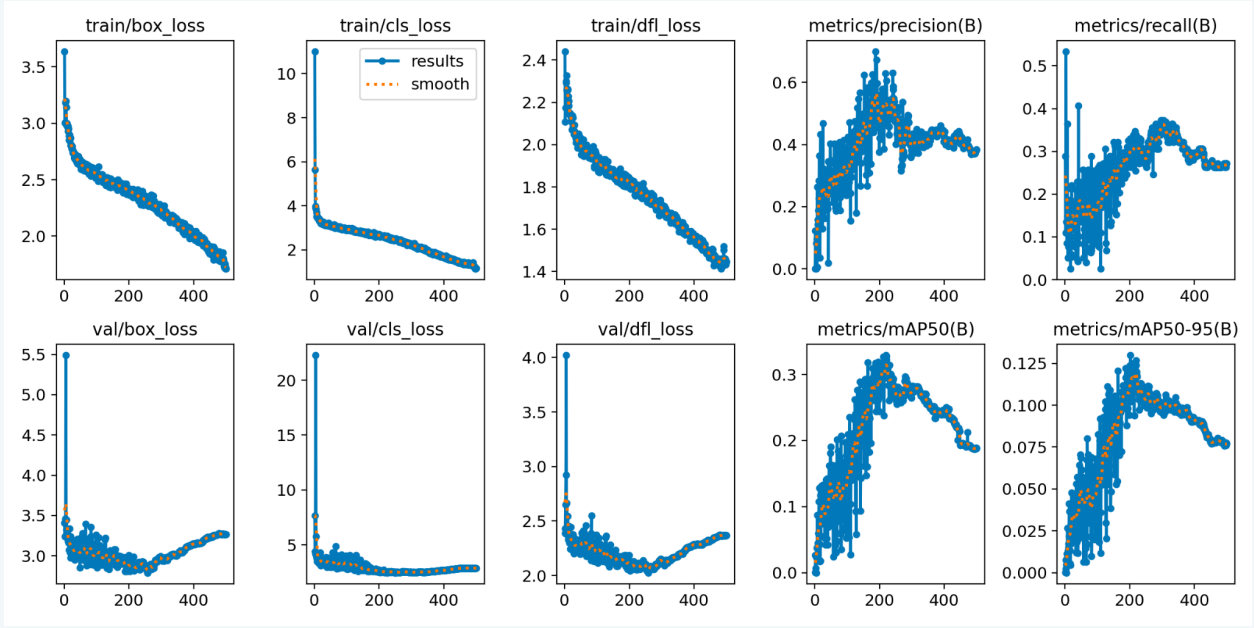
This model is trained with dropout=0.2, learning_rate = 0.001, with image resized to 640 by 640, for around 150 epochs which took around 45 minutes with gpu.




metrics/precision(B)
metrics/recall(B)
metrics/mAP50(B)
metrics/mAP50-95(B)


Precision-Recall Curve
tumor 0.275
all classes 0.275 mAP@0.5
Precision
Recall

We trained another model with various data augmention techniques like horizontal and vertical flipping, random image cropping, higher dropout = 0.5 and larger image size(1000 by 1000), for 500 epochs which took around 3 hours with gpu. Below are the metrics associated with training this model.



The results were similar to the first one so we stuck to that.

Results:

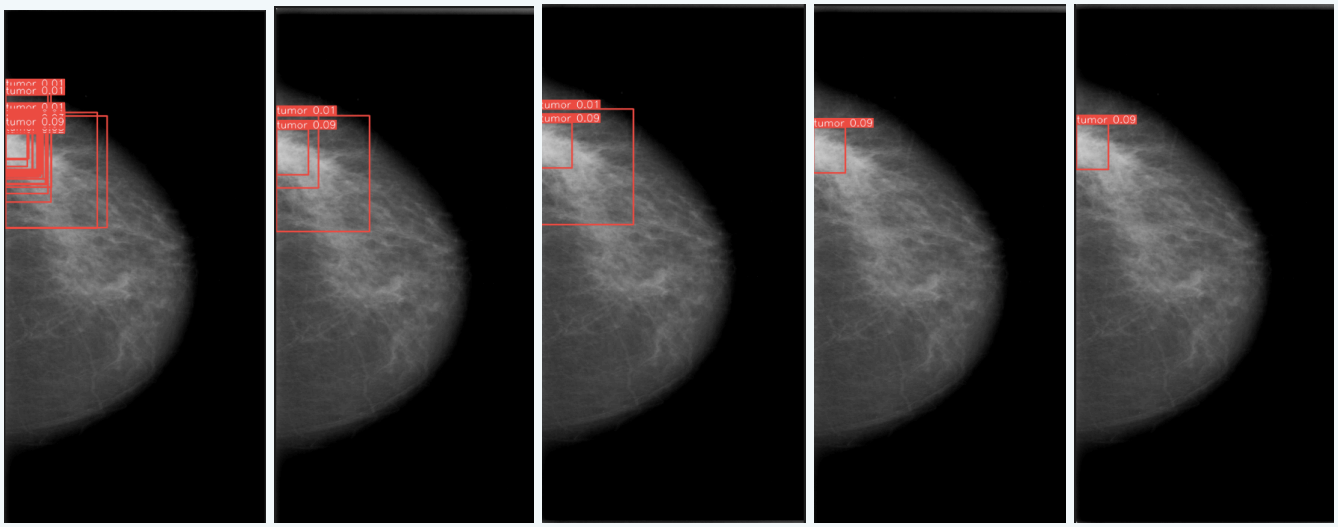
Froc.py output:

```
FPI: 0.0, Sensitivity: 0.05224, Threshold: 0.48348
FPI: 0.1, Sensitivity: 0.06716, Threshold: 0.38438
FPI: 0.1, Sensitivity: 0.11194, Threshold: 0.23223
FPI: 0.1, Sensitivity: 0.13433, Threshold: 0.16917
FPI: 0.2, Sensitivity: 0.14179, Threshold: 0.13914
FPI: 0.3, Sensitivity: 0.17164, Threshold: 0.08008
FPI: 0.4, Sensitivity: 0.19403, Threshold: 0.05405
FPI: 0.5, Sensitivity: 0.23881, Threshold: 0.03203
FPI: 0.6, Sensitivity: 0.24627, Threshold: 0.02302
FPI: 0.7, Sensitivity: 0.25373, Threshold: 0.01802
FPI: 0.8, Sensitivity: 0.25373, Threshold: 0.01401
FPI: 1.0, Sensitivity: 0.26119, Threshold: 0.01001
FPI: 1.1, Sensitivity: 0.26866, Threshold: 0.00801
FPI: 1.2, Sensitivity: 0.28358, Threshold: 0.00701
FPI: 1.5, Sensitivity: 0.29104, Threshold: 0.00501
FPI: 1.8, Sensitivity: 0.29104, Threshold: 0.004
FPI: 1.9, Sensitivity: 0.30597, Threshold: 0.003
FPI: 2.0, Sensitivity: 0.30597, Threshold: 0.003
FPI: 2.4, Sensitivity: 0.31343, Threshold: 0.002
FPI: 2.7, Sensitivity: 0.31343, Threshold: 0.002
FPI: 3.0, Sensitivity: 0.31343, Threshold: 0.002
FPI: 4.4, Sensitivity: 0.45522, Threshold: 0.001
FPI: 5.4, Sensitivity: 0.45522, Threshold: 0.001
```

Impact of NMS on the output:

image_id: 0 in coco/val
Image_name: Calc-Training_P_01250_LEFT_CC.png

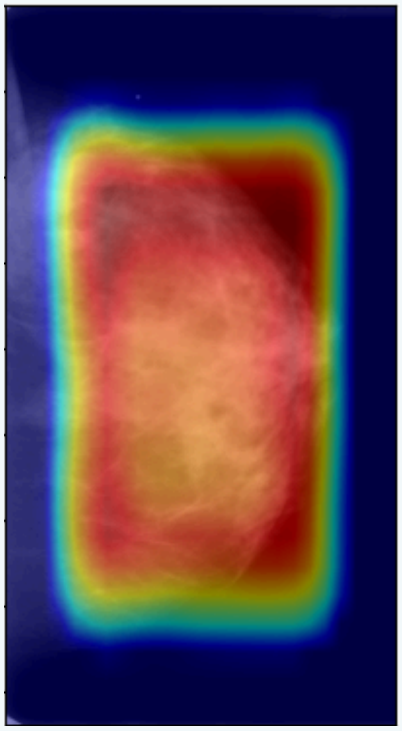
The Red box in the above image shows the ground truth bounding box.



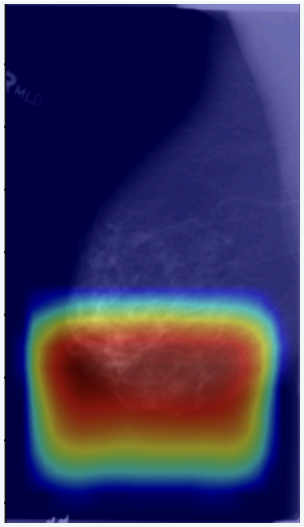
From Left to Right IoU thresholds: 1, 0.5, 0.25, 0.1, and 0

Example Gradcams:

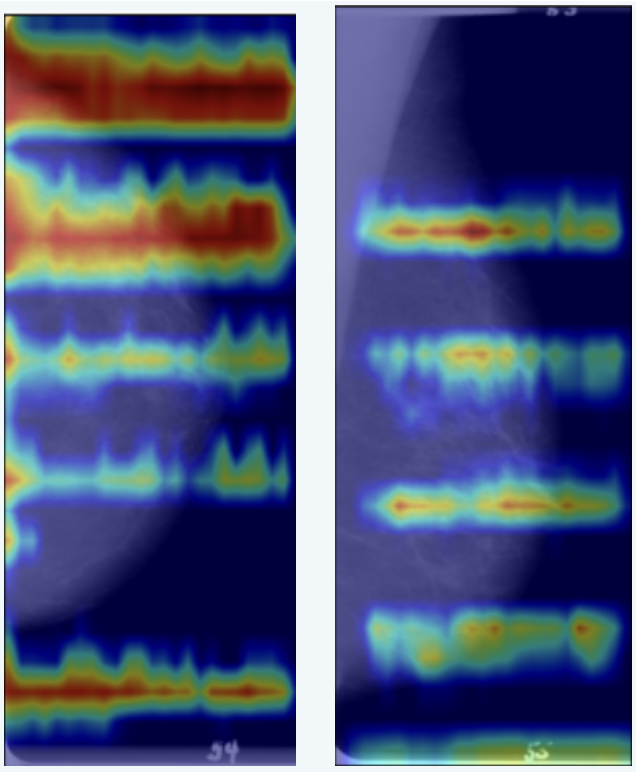
image_name : Calc-Training_P_00048_RIGHT_CC.png in sample test dir:



image_name : Calc-Training_P_00349_RIGHT_MLO.png in sample test dir:



On malignant images visualisation of gradcams:

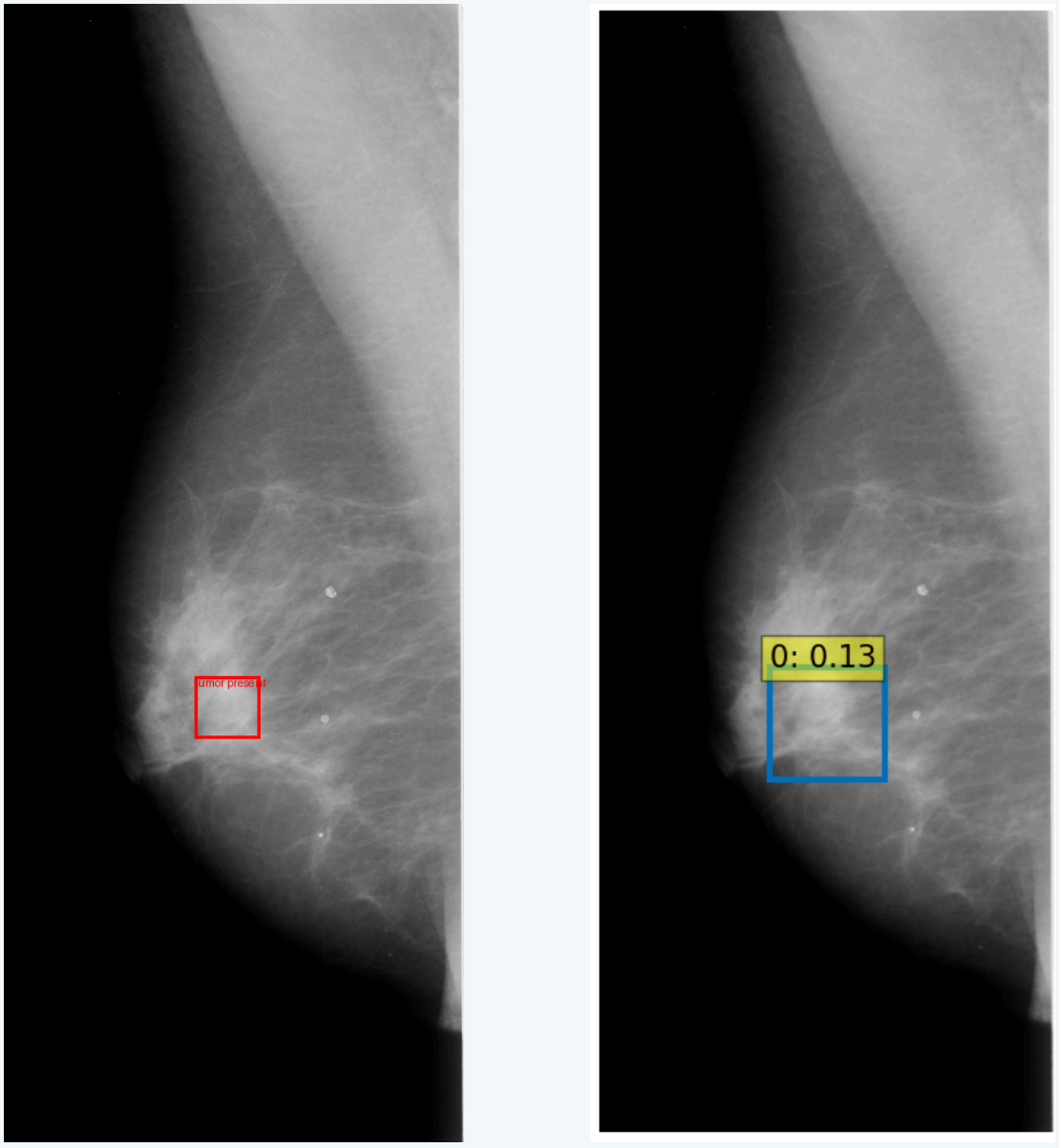


We can clearly see through the heatmaps, that there is something clearly wrong with the model's prediction on these images.

2) **Transformer-based DETR model:**

Data Visualisations:

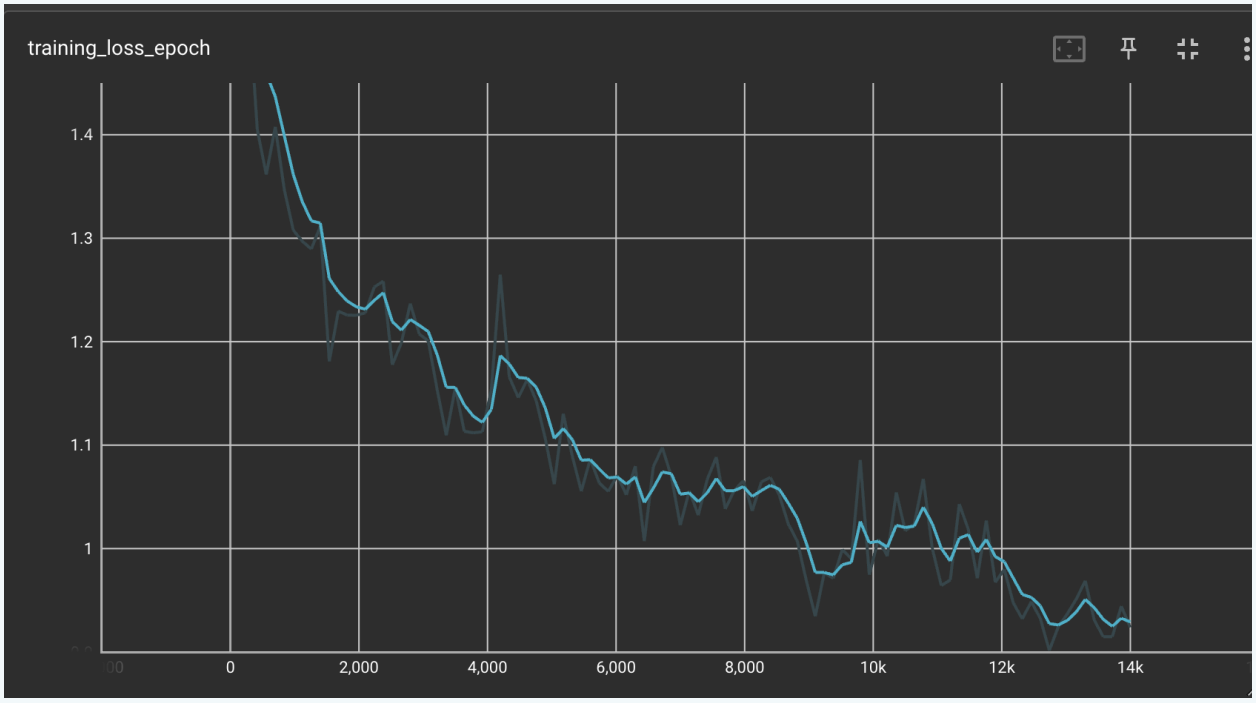
Image_id:7 in coco/train folder



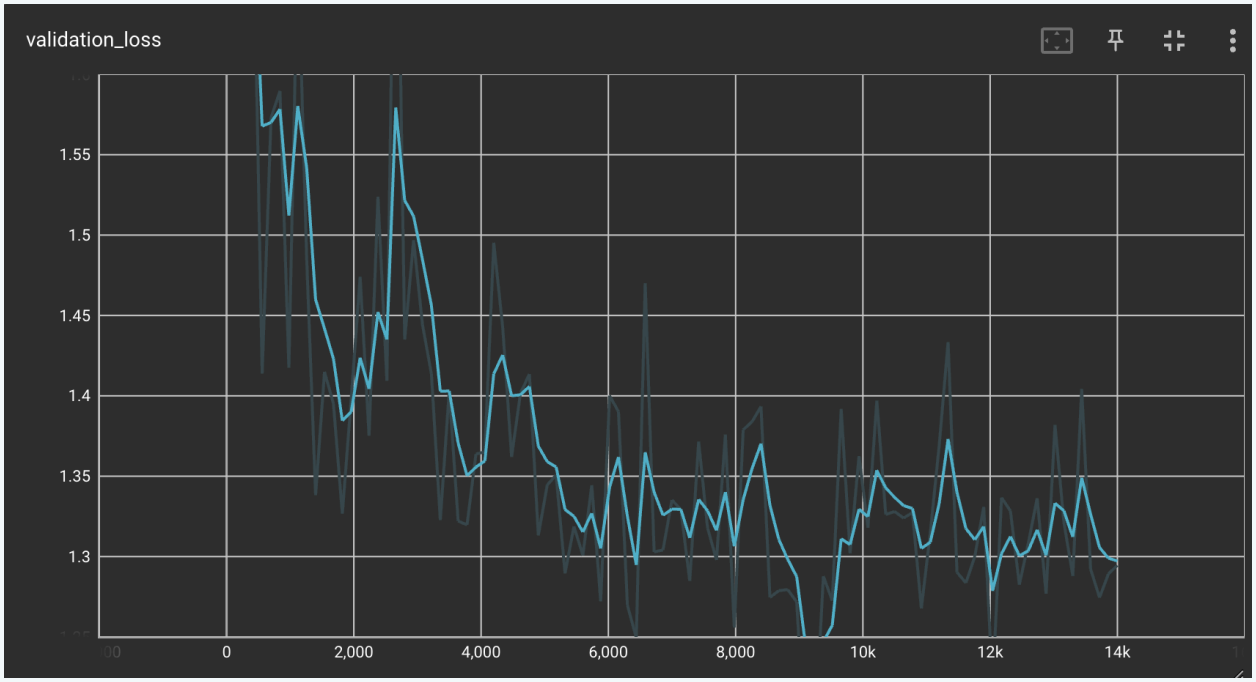
(On Left: Label Bounding Box, On Right: Predicted Bounding Box)

PLOTS:

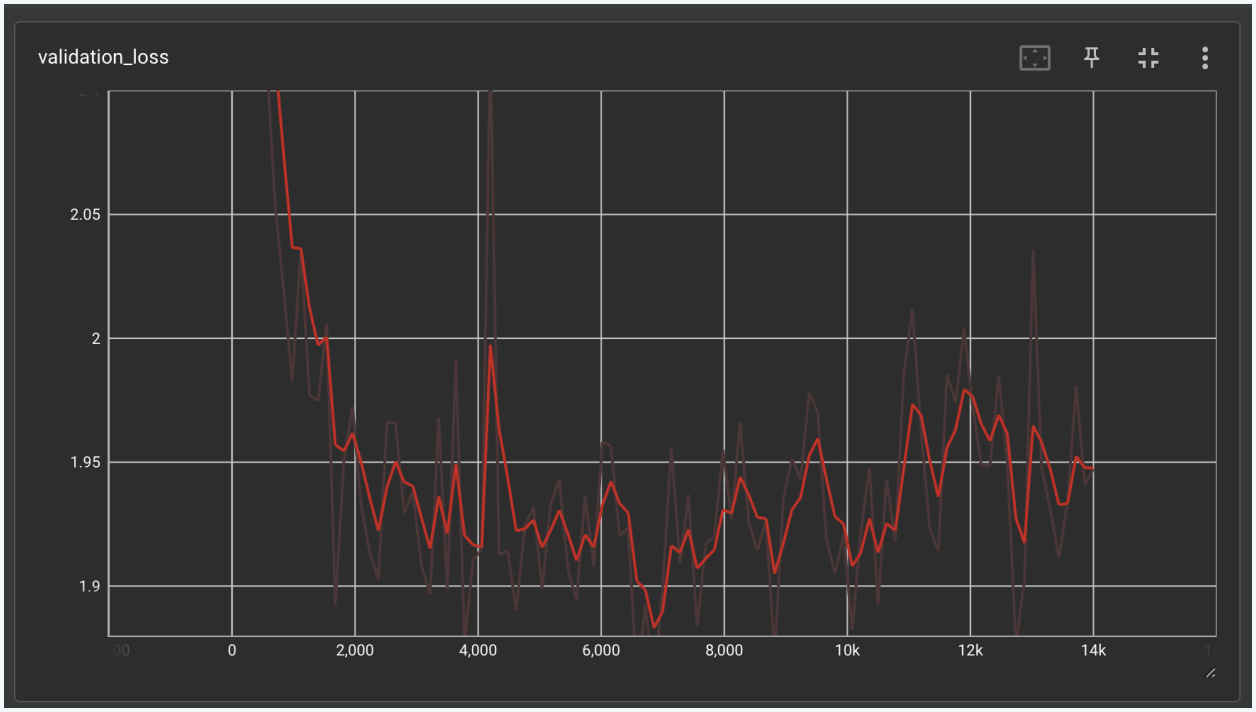
Training Loss:



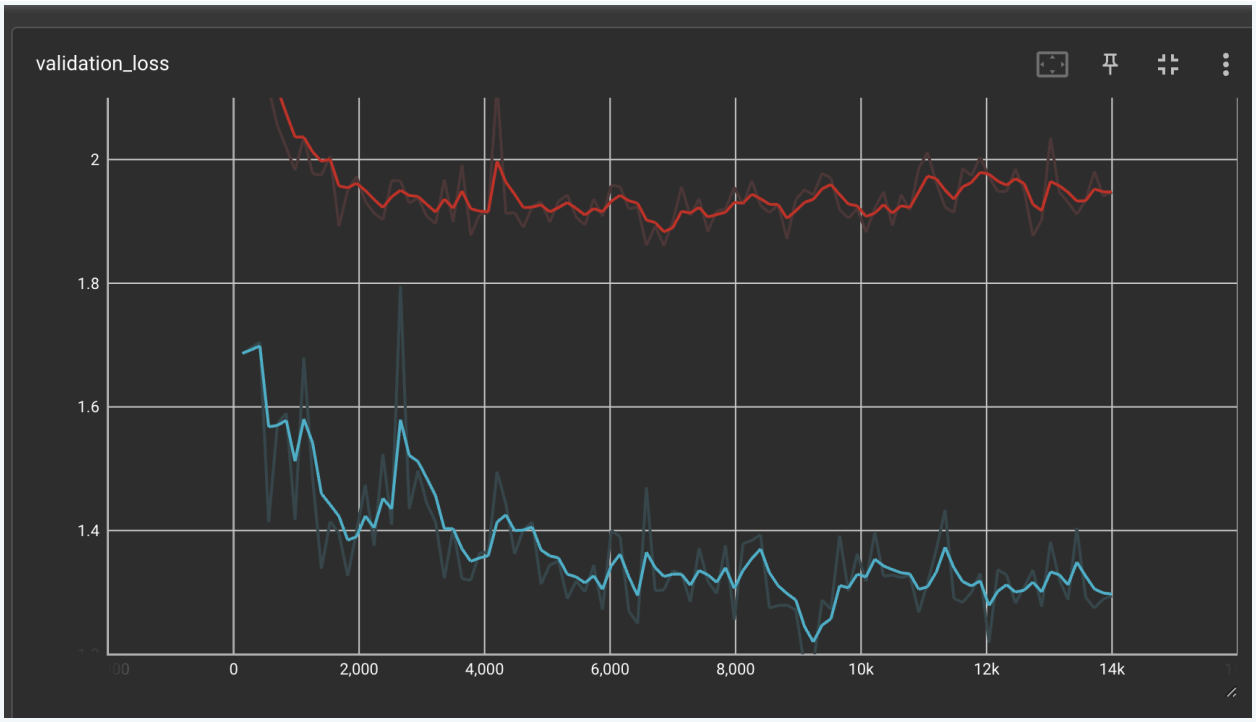
Validation Loss



We had also tried preprocessing the data such that all the images which didn't have any tumor, instead of them having empty annotations, I added a class label of 1 and bounding boxes as a tensor of (0,0,0,0). The results of the model are:



Compared to the results without preprocessing:



The blue one represents the model without this preprocessing and the red one with it.

We can clearly observe that without this preprocessing step the model has less loss and hence better performance. This could be because adding another label and dummy bounding boxes adds unnecessary complexity to the problem which leads to worse performance.

Results:

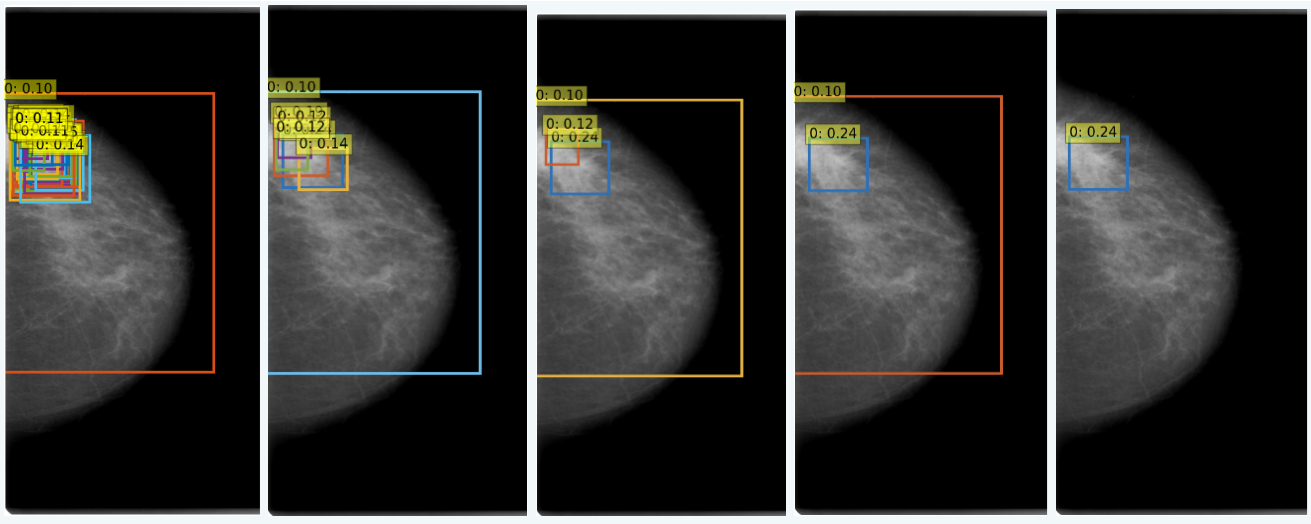
Froc.py output:

```
FPI: 0.0, Sensitivity: 0.20896, Threshold: 0.23023
FPI: 0.1, Sensitivity: 0.24627, Threshold: 0.21522
FPI: 0.1, Sensitivity: 0.29851, Threshold: 0.16917
FPI: 0.1, Sensitivity: 0.33582, Threshold: 0.14515
FPI: 0.2, Sensitivity: 0.37313, Threshold: 0.12913
FPI: 0.3, Sensitivity: 0.44776, Threshold: 0.10811
FPI: 0.4, Sensitivity: 0.46269, Threshold: 0.0961
FPI: 0.5, Sensitivity: 0.46269, Threshold: 0.08809
FPI: 0.6, Sensitivity: 0.47761, Threshold: 0.08008
FPI: 0.7, Sensitivity: 0.48507, Threshold: 0.07608
FPI: 0.8, Sensitivity: 0.5, Threshold: 0.07207
FPI: 1.0, Sensitivity: 0.53731, Threshold: 0.06507
FPI: 1.1, Sensitivity: 0.56716, Threshold: 0.06206
FPI: 1.2, Sensitivity: 0.57463, Threshold: 0.05906
FPI: 1.5, Sensitivity: 0.59701, Threshold: 0.05105
FPI: 1.8, Sensitivity: 0.61194, Threshold: 0.04505
FPI: 1.9, Sensitivity: 0.61194, Threshold: 0.04404
FPI: 2.0, Sensitivity: 0.6194, Threshold: 0.04204
FPI: 2.4, Sensitivity: 0.62687, Threshold: 0.03804
FPI: 2.7, Sensitivity: 0.66418, Threshold: 0.03403
FPI: 3.0, Sensitivity: 0.66418, Threshold: 0.03203
FPI: 4.4, Sensitivity: 0.70149, Threshold: 0.02402
FPI: 5.4, Sensitivity: 0.70896, Threshold: 0.02102
```

Impact of NMS on the output:

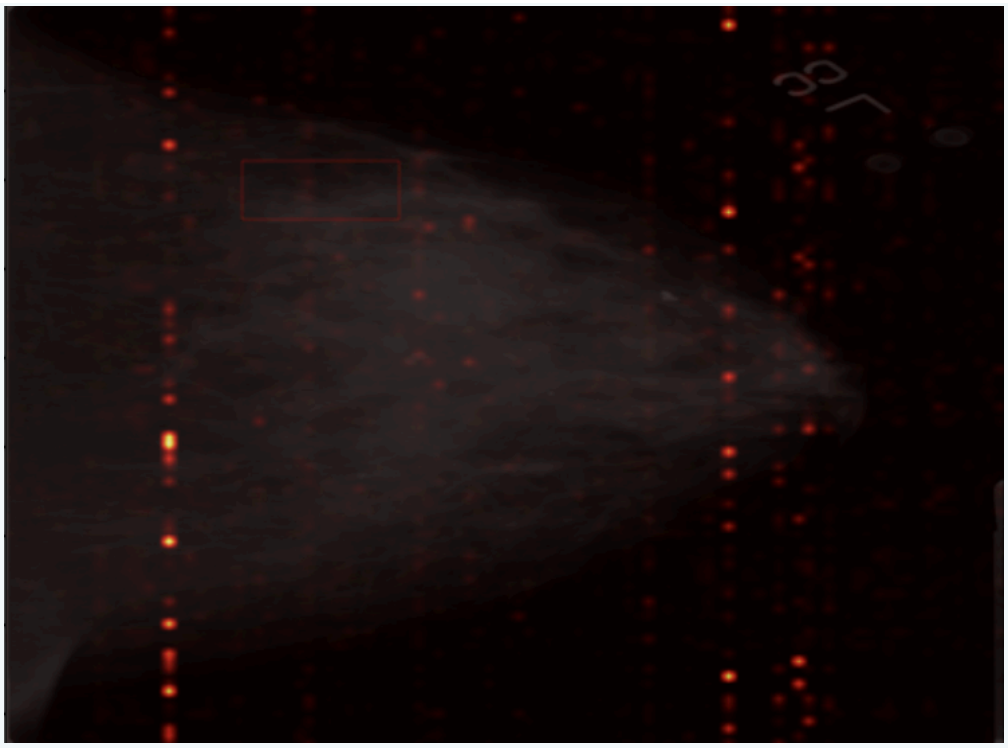
image_id: 0 in coco/val

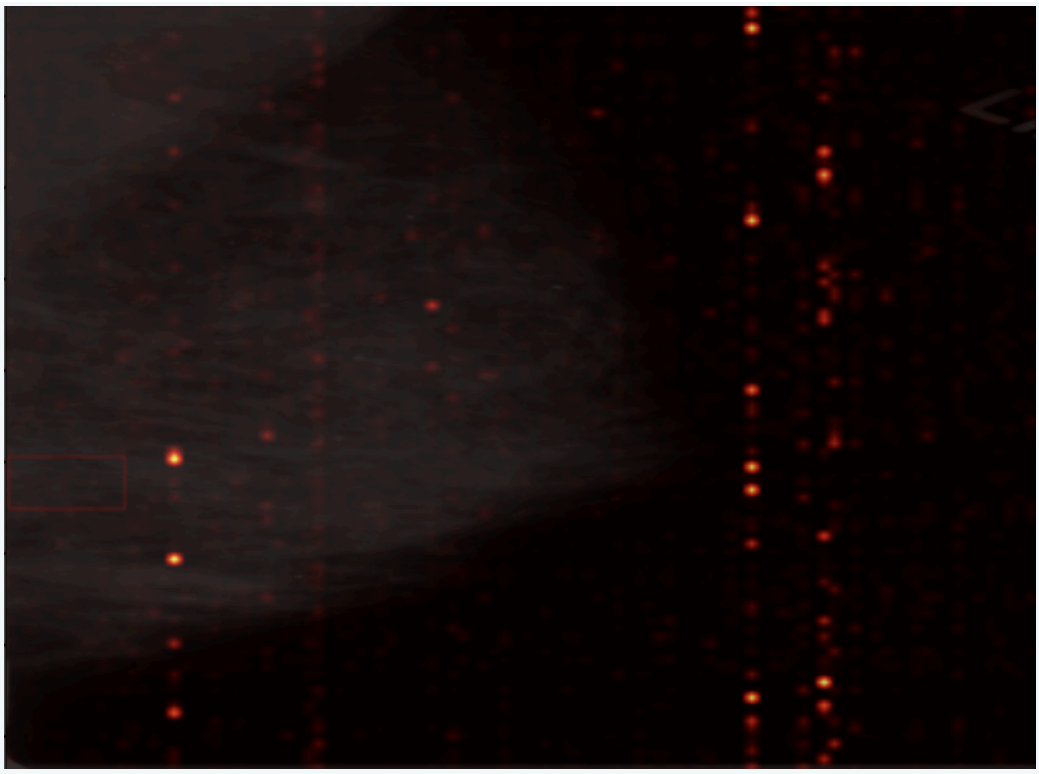
The Red box in the above image shows the ground truth bounding box.



From Left to Right IoU thresholds: 1, 0.5, 0.25, 0.1, and 0

Example Attention Maps:



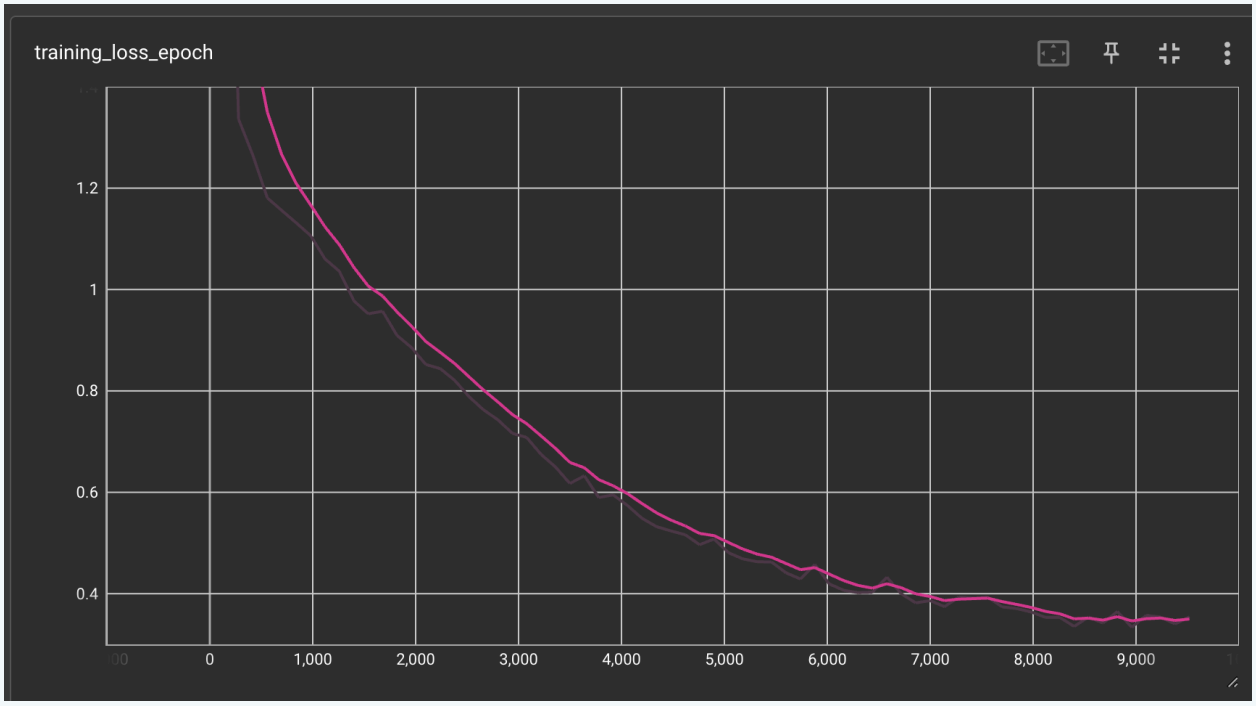


Malignant Images:

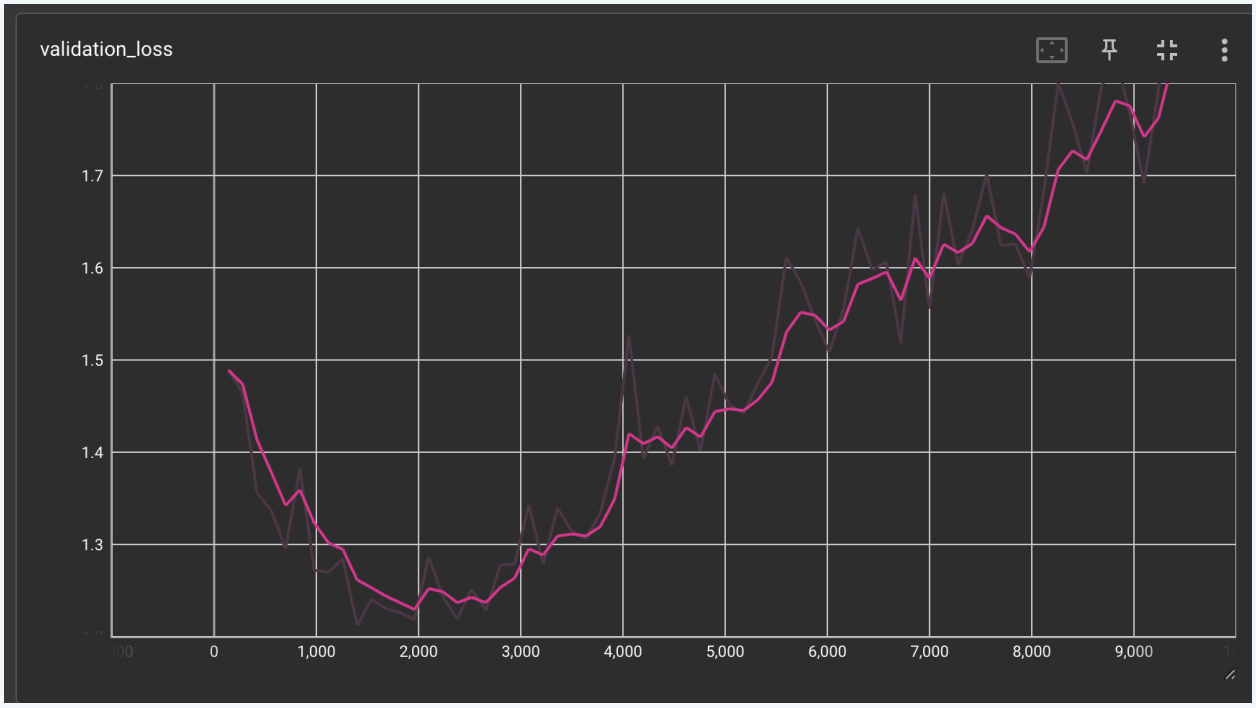
Experimentation:

We had also tried training the Deformable-Detr, but it was not giving any better results than the Detr model:

Training Loss:



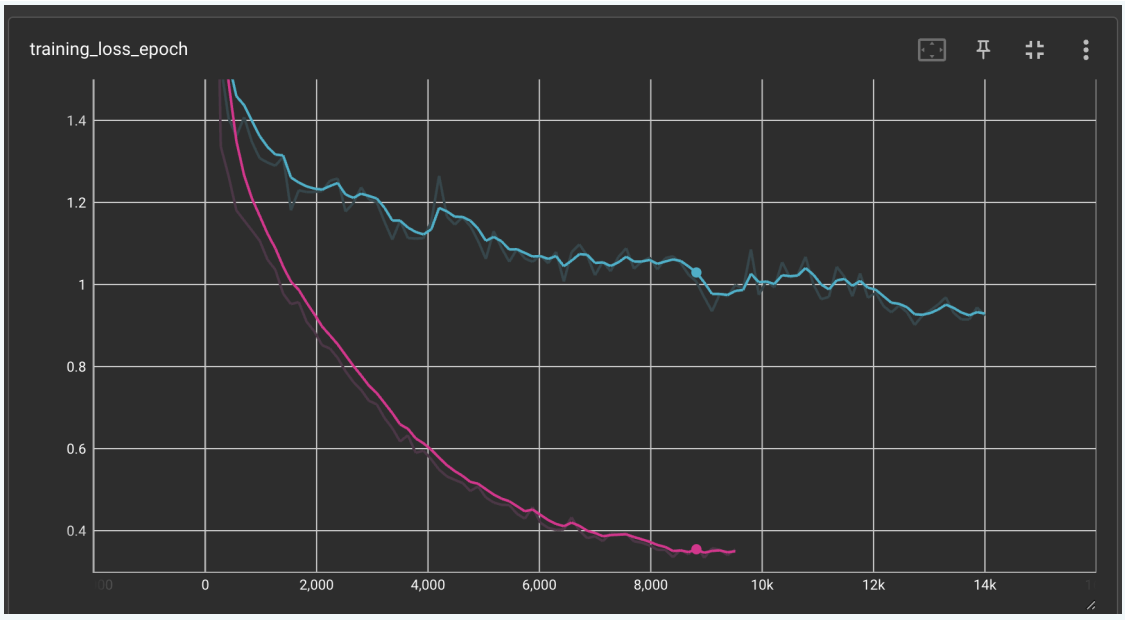
Validation Loss



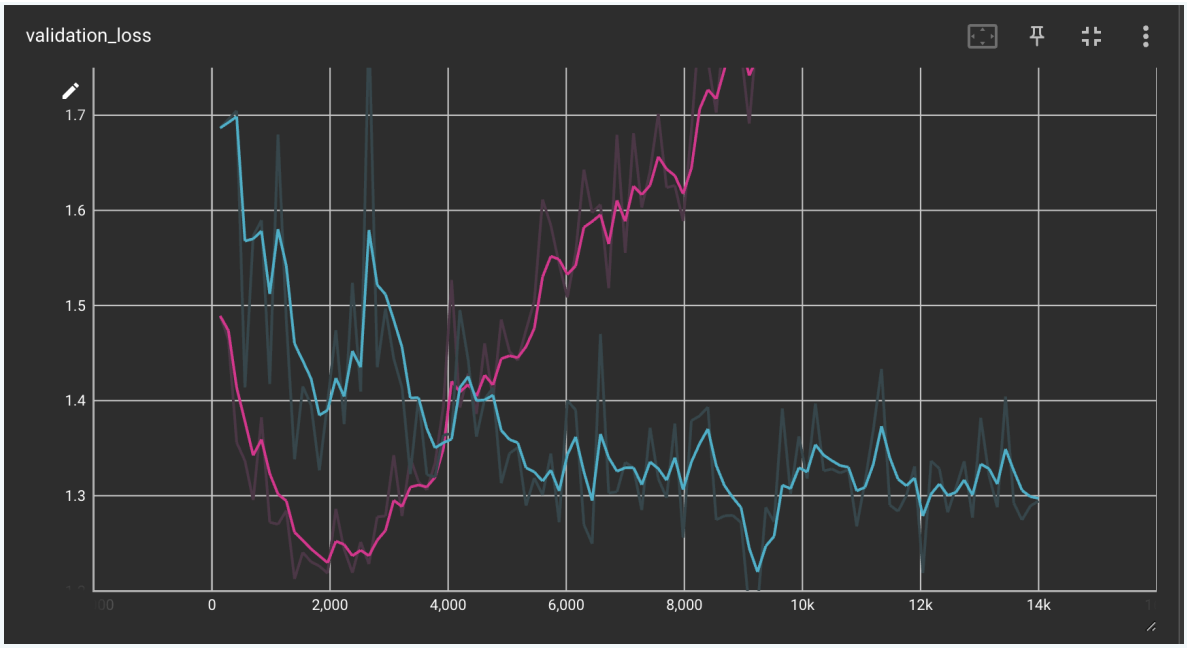
We observed that the Deformable Detr model overfitted very quickly. Though it gave pretty low training loss value in very few iterations.

Detr vs Deformable Detr:

Training Loss:



Validation Loss:



The Blue line is of the Detr and pink line is of the Deformable Detr.

We observe that although the Deformable Detr converged much faster and therefore had lesser training loss than the Detr model, but such fast convergence led to it overfitting on the dataset and thus the deformable detr showed higher validation losses than the Detr model. So we stuck with using the Detr model only.

Comparison of Yolo vs Transformer based Detr model :

```
FPI: 0.0, Sensitivity: 0.20896, Threshold: 0.23023
FPI: 0.1, Sensitivity: 0.24627, Threshold: 0.21522
FPI: 0.1, Sensitivity: 0.29851, Threshold: 0.16917
FPI: 0.1, Sensitivity: 0.33582, Threshold: 0.14515
FPI: 0.2, Sensitivity: 0.37313, Threshold: 0.12913
FPI: 0.3, Sensitivity: 0.44776, Threshold: 0.10811
FPI: 0.4, Sensitivity: 0.46269, Threshold: 0.0961
FPI: 0.5, Sensitivity: 0.46269, Threshold: 0.08809
FPI: 0.6, Sensitivity: 0.47761, Threshold: 0.08008
FPI: 0.7, Sensitivity: 0.48507, Threshold: 0.07608
FPI: 0.8, Sensitivity: 0.5, Threshold: 0.07207
FPI: 1.0, Sensitivity: 0.53731, Threshold: 0.06507
FPI: 1.1, Sensitivity: 0.56716, Threshold: 0.06206
FPI: 1.2, Sensitivity: 0.57463, Threshold: 0.05906
FPI: 1.5, Sensitivity: 0.59701, Threshold: 0.05105
FPI: 1.8, Sensitivity: 0.61194, Threshold: 0.04505
FPI: 1.9, Sensitivity: 0.61194, Threshold: 0.04404
FPI: 2.0, Sensitivity: 0.6194, Threshold: 0.04204
FPI: 2.4, Sensitivity: 0.62687, Threshold: 0.03804
FPI: 2.7, Sensitivity: 0.66418, Threshold: 0.03403
FPI: 3.0, Sensitivity: 0.66418, Threshold: 0.03203
FPI: 4.4, Sensitivity: 0.70149, Threshold: 0.02402
FPI: 5.4, Sensitivity: 0.70896, Threshold: 0.02102
```

Transformer

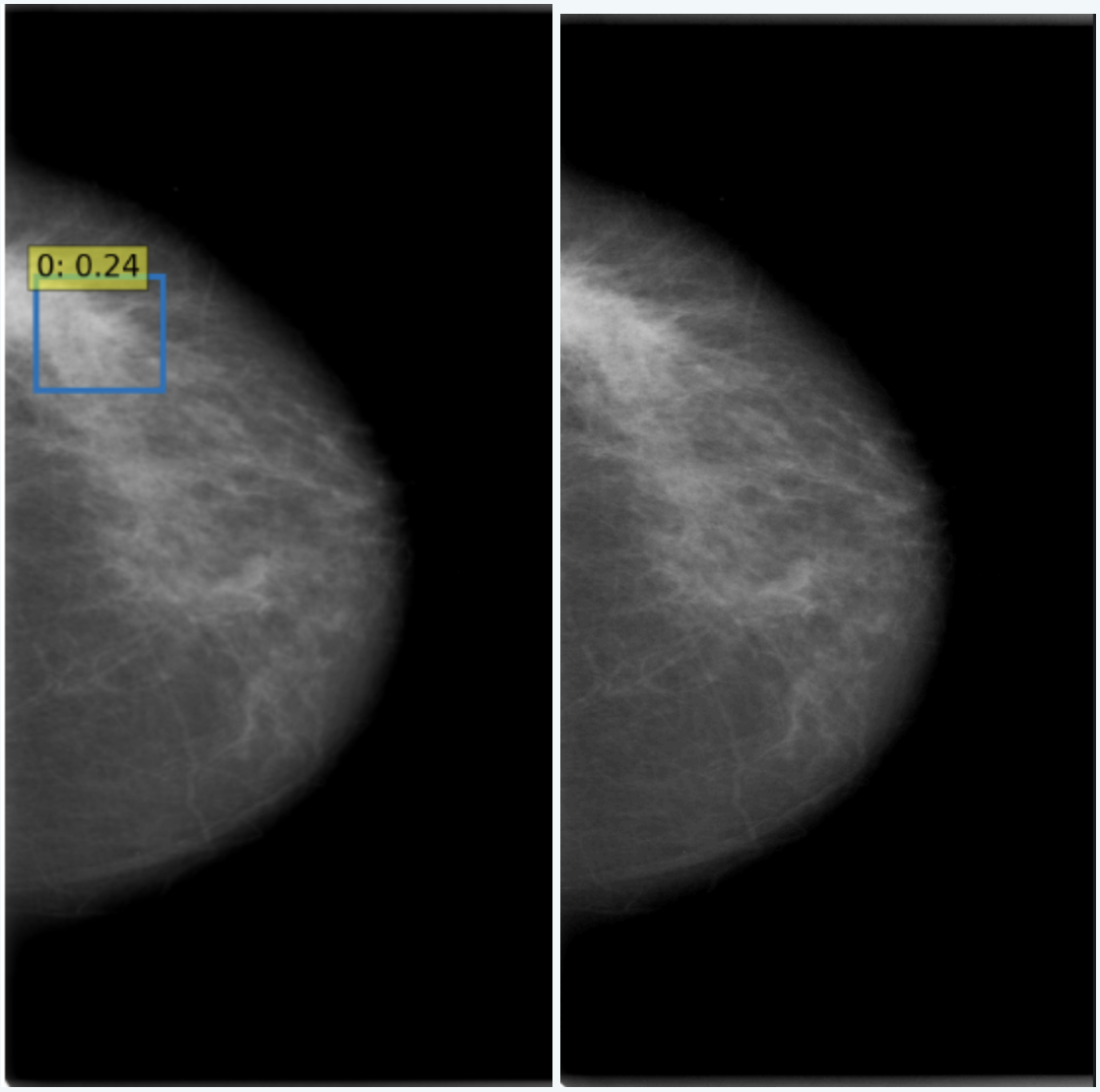
```
FPI: 0.0, Sensitivity: 0.05224, Threshold: 0.48348
FPI: 0.1, Sensitivity: 0.06716, Threshold: 0.38438
FPI: 0.1, Sensitivity: 0.11194, Threshold: 0.23223
FPI: 0.1, Sensitivity: 0.13433, Threshold: 0.16917
FPI: 0.2, Sensitivity: 0.14179, Threshold: 0.13914
FPI: 0.3, Sensitivity: 0.17164, Threshold: 0.08008
FPI: 0.4, Sensitivity: 0.19403, Threshold: 0.05405
FPI: 0.5, Sensitivity: 0.23881, Threshold: 0.03203
FPI: 0.6, Sensitivity: 0.24627, Threshold: 0.02302
FPI: 0.7, Sensitivity: 0.25373, Threshold: 0.01802
FPI: 0.8, Sensitivity: 0.25373, Threshold: 0.01401
FPI: 1.0, Sensitivity: 0.26119, Threshold: 0.01001
FPI: 1.1, Sensitivity: 0.26866, Threshold: 0.00801
FPI: 1.2, Sensitivity: 0.28358, Threshold: 0.00701
FPI: 1.5, Sensitivity: 0.29104, Threshold: 0.00501
FPI: 1.8, Sensitivity: 0.29104, Threshold: 0.004
FPI: 1.9, Sensitivity: 0.30597, Threshold: 0.003
FPI: 2.0, Sensitivity: 0.30597, Threshold: 0.003
FPI: 2.4, Sensitivity: 0.31343, Threshold: 0.002
FPI: 2.7, Sensitivity: 0.31343, Threshold: 0.002
FPI: 3.0, Sensitivity: 0.31343, Threshold: 0.002
FPI: 4.4, Sensitivity: 0.45522, Threshold: 0.001
FPI: 5.4, Sensitivity: 0.45522, Threshold: 0.001
```

Yolo

So we observe that the transformer based model has higher sensitivity values than yolo one corresponding to the same FPI values. This means that the transformer detect more true positives than the yolo model at the same false positive detection rate.

Let us select an FPI value of 0.2 and compare the predictions of both the models at the corresponding thresholds.

On image_id=0 in coco/val, image_name: Calc-Training_P_01250_LEFT_CC.png

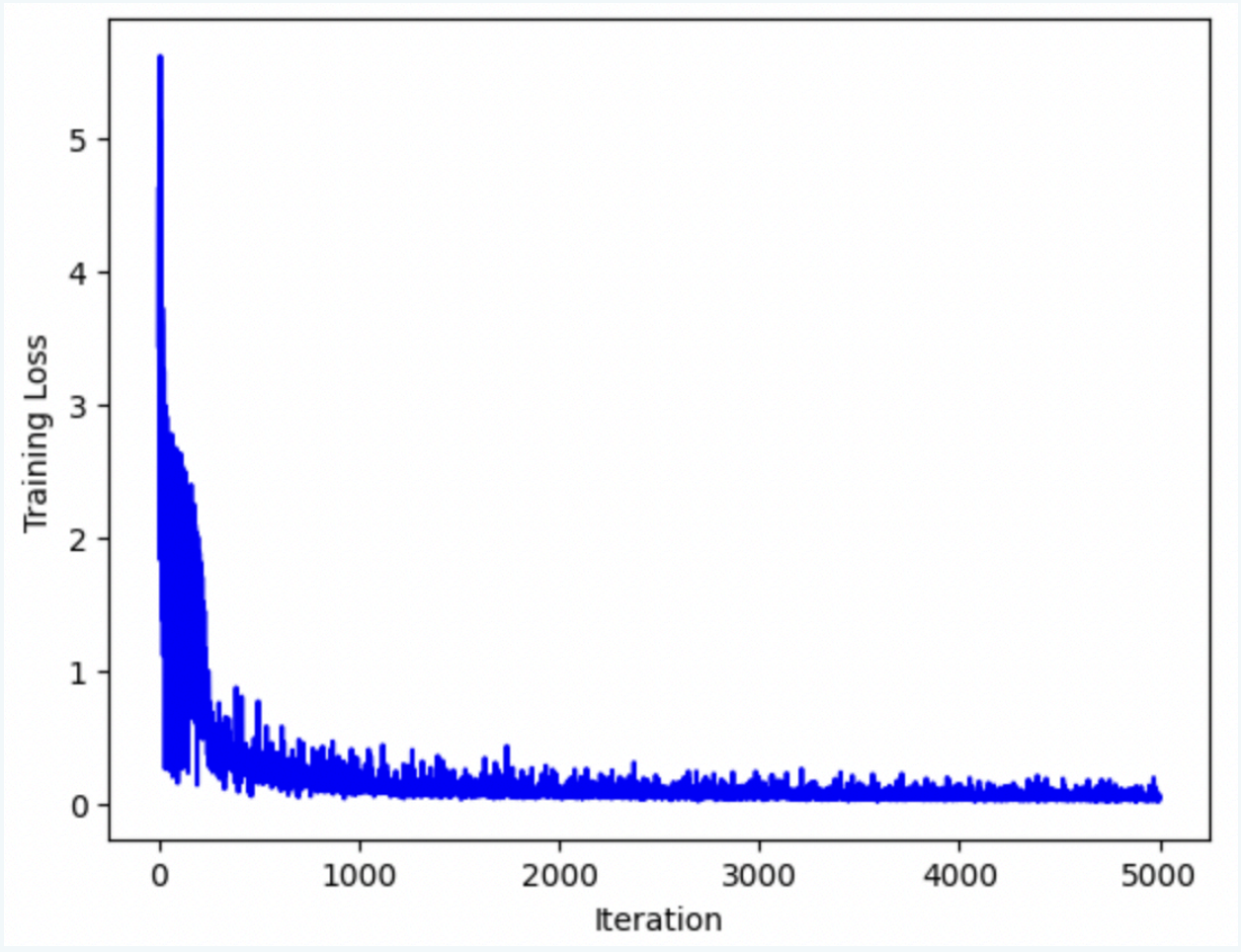


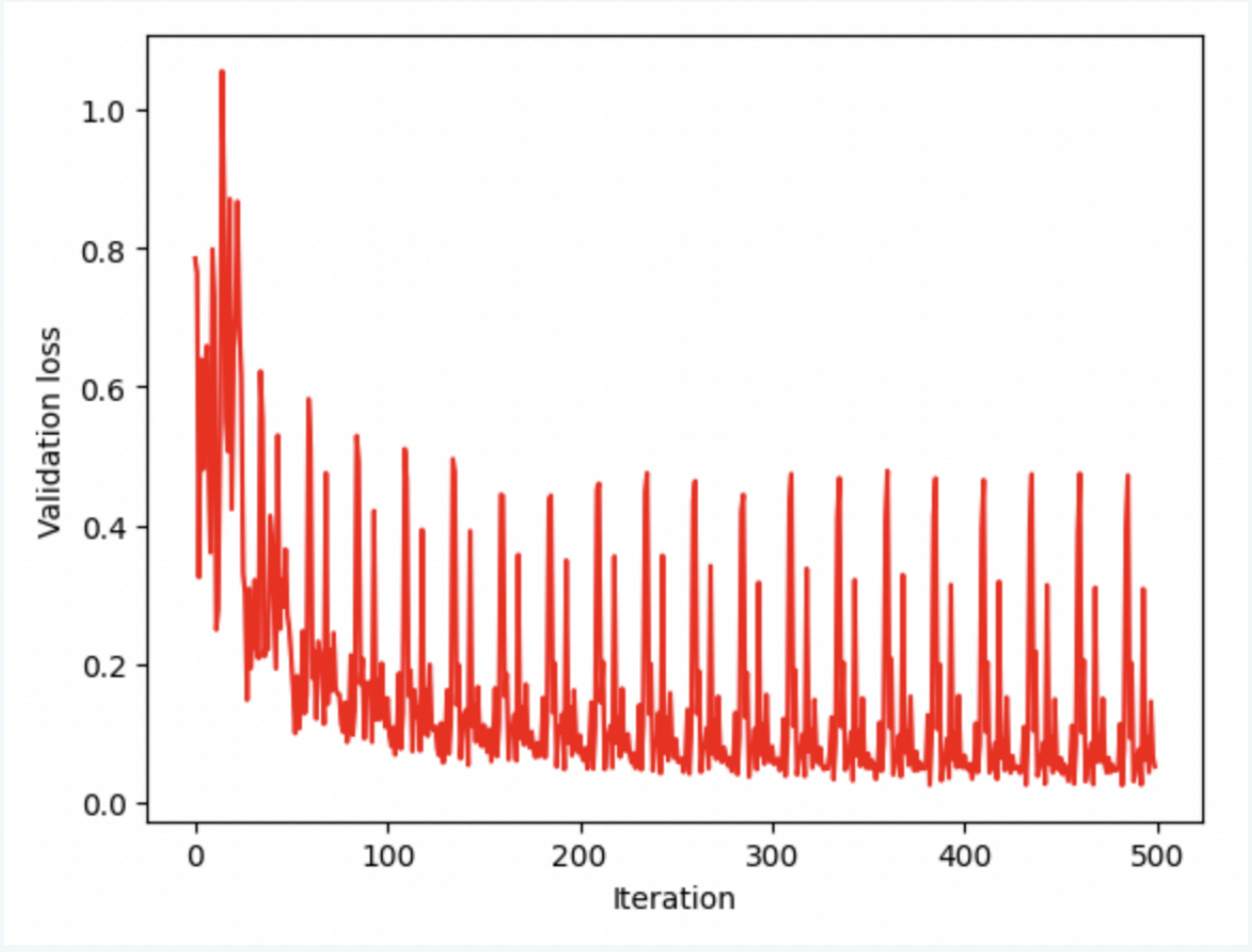
On Left: Detr output On Right: Yolo Output (No Detection)

We observe that at these thresholds, the Detr model is able to predict the tumor and also locate it correctly, whereas the Yolo model is unable to even detect the tumor. Hence the FPI values and their corresponding sensitivity values make sense and we are able to verify these results visually on the dataset as well.

3. **RCNN based model**

We also implemented the FasterRCNN based object detection model. Here are the training loss and validation loss plots.

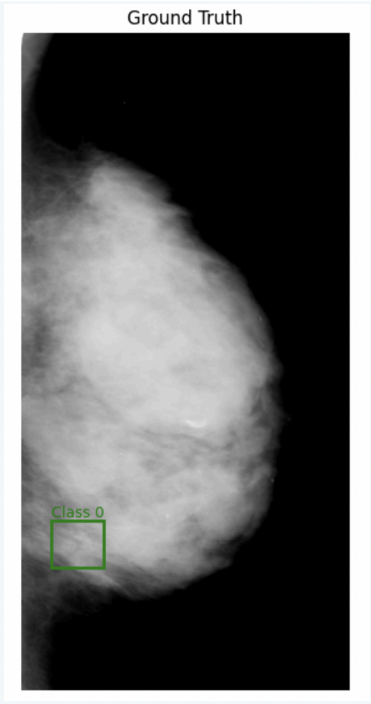




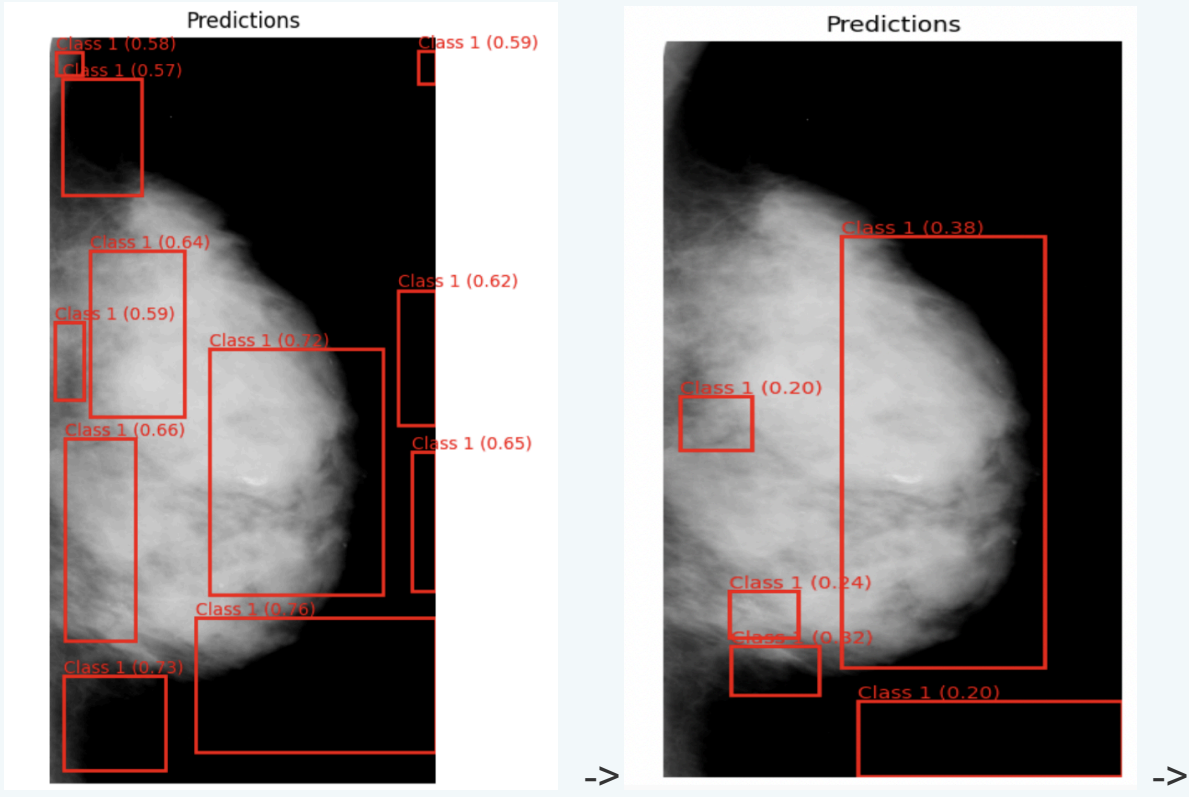
As you can see the training loss decreases down to 0 as the iterations keep increasing

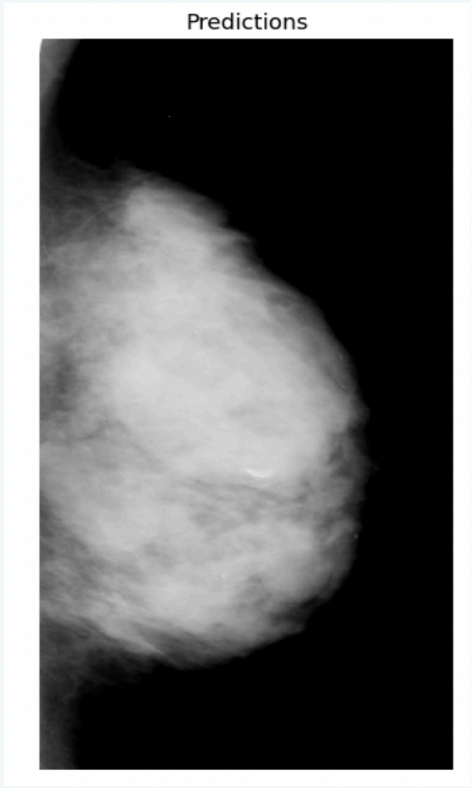
You can see here that with the increasing epoch, the predicted boxes are decreasing. After a certain number of epochs, for almost all the images, the boxes predicted were nil, and the loss reported was 0. This means that the model finds it the most optimal to report no boxes and converges.

Example image



Below we have shown how the predictions kept on decreasing and finally there were no predictions


Predictions
Class 1 (0.58)
Class 1 (0.57)
Class 1 (0.59)
Class 1 (0.64)
Class 1 (0.62)
Class 1 (0.59)
Class 1 (0.72)
Class 1 (0.66)
Class 1 (0.65)
Class 1 (0.76)
Class 1 (0.73)
->
Predictions
Class 1 (0.38)
Class 1 (0.20)
Class 1 (0.24)
Class 1 (0.32)
Class 1 (0.20)
->
Predictions
Class 1 (0.13)
Class 1 (0.06)
->
Predictions
Class 1 (0.12)
->



You can see that there are no predictions.

- All the codes that we have tried out (which we later rejected due to bad performance), can be found in the experiments folder.

Link to Saved Model Checkpoint: https://drive.google.com/drive/folders/1o1JS_3H4WNb3tIBXSGurSuZAKGL6ujJz?usp=sharing