**Deep learning in mammography to detect abnormality breast cancer tissue**

**Yixin Huang, Canaan Law,** [**Rushikesh Jagtap**](https://sjsu.instructure.com/courses/1485410/users/4488369)

**Project Summary**

Breast cancer is one of the most common cancers that happen to women. The rule for diagnosis is based on the analysis of mammographic images, but the accuracy is always limited by doctors’ experience and skills. In this paper, we will explore multiple neural networks models to improve the detection accuracy, and compare the performances among these approaches.

**Problem definition**

Computer-assisted detection (CAD) has been commonly used in mammography for decades. However it is still not a perfect solution after years of implementation. Some studies pointed out that an earlier detection would improve the patients severity of breast cancers. Also, CAD still involves a decent amount of false positives in terms of detection, which is a serious problem for patients. We would like to apply a deep learning method on mammography images to detect problematic areas.

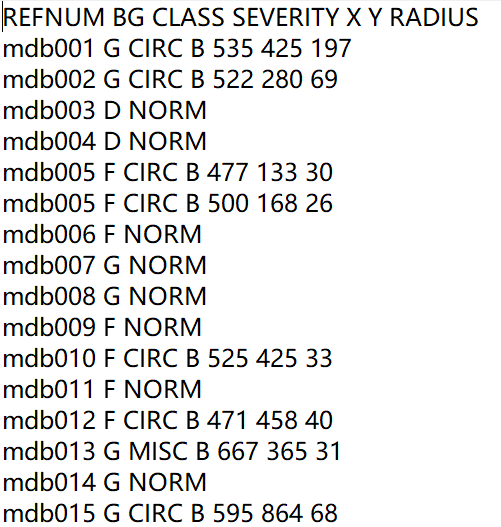
**Approach to problem:**

We plan to extract images from the mini MIAS database. Since the bounding boxes are using x, y, and radius, we need to pre-process the coordinate to YOLO format. We plan to use some data augmentation techniques for removing noise from images along with horizontal flipping, zooming, etc. We plan to have less amount of images for validation and more for training data as we have a smaller dataset. The model we plan to implement is YOLOv5 and Retinanet. We also plan to evaluate the model on various metrics .

**Dataset**

The mini-MIAS database of mammograms contained 322 Images and info files that contained other details. Such details included reference number, character of background tissue, class of abnormality, severity of abnormality, x and y coordinates of center of abnormality, and the radius. If the image is classified as abnormal, then the image has the corresponding coordinate. The size of all the images is 1024 x 1024 pixels. And the coordinate system originates at the bottom left corner.

Figure 1: Original Data Format



**Preprocessing**

In order to train with Yolov5 and Retinanet, there are a few preprocessing steps that need to be done. Firstly, we read the txt file that contains all the labels and coordinate them into a dataframe. We then remove all the invalid data. There are a few images classified as abnormal, but without coordinates. We also change categorical labels to numeric values to match with the yolo bounding box requirement.

The second important preprocessing process is converting the x,y, and radius value to bounding boxes. The bounding box x\_min, y\_min, x\_max, and y max can be calculated by the provided value. For example, the formula of x\_min is x - radius, and x\_max is x + radius. However, we need an extra step for converting y\_min and y\_max due to the coordinate system being different between the original image and normal bounding box system. The y-axis is flipped. Instead of using the same formula for x-axis, the y\_min and y\_max formula is image height - y +/- radius. This will form a bounding of the outside of the circle. Once we have all the bounding box values, we can write it into a txt file and use it for Retinanet or convert to yolo format for yolov5.

Figure 2: Explanation of COCO and YOLO format bounding box

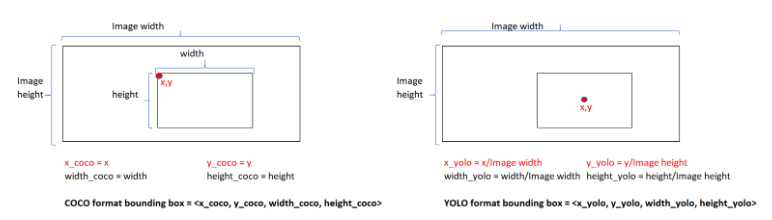


Figure 3: Formulas of Circumscribed Bounding Box YOLO Format Converting

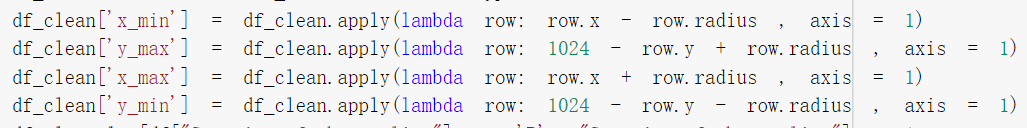


Figure 4: Formulas of YOLO Circumscribed Bounding Box

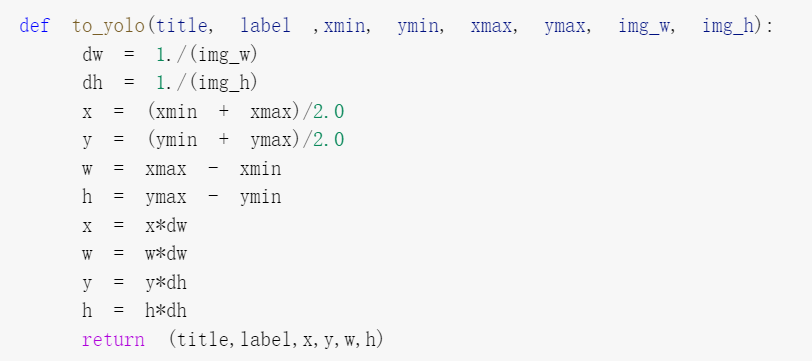


Figure 5: Formulas of YOLO Inscribed Bounding Box

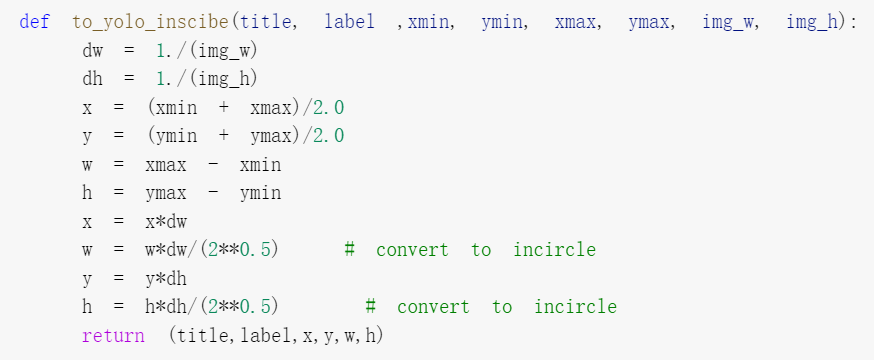
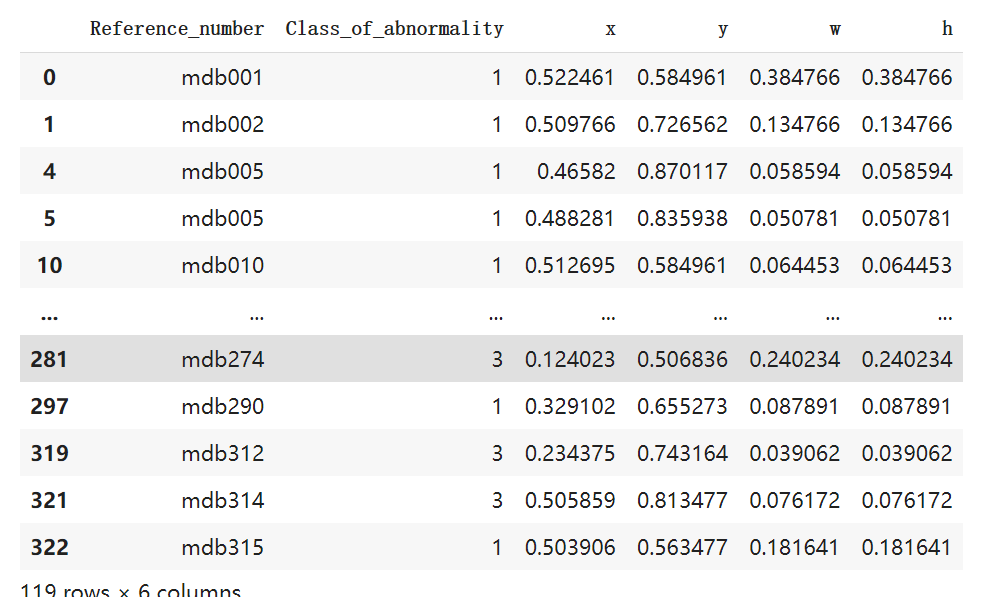


Figure 6: Final Format for YOLOv5

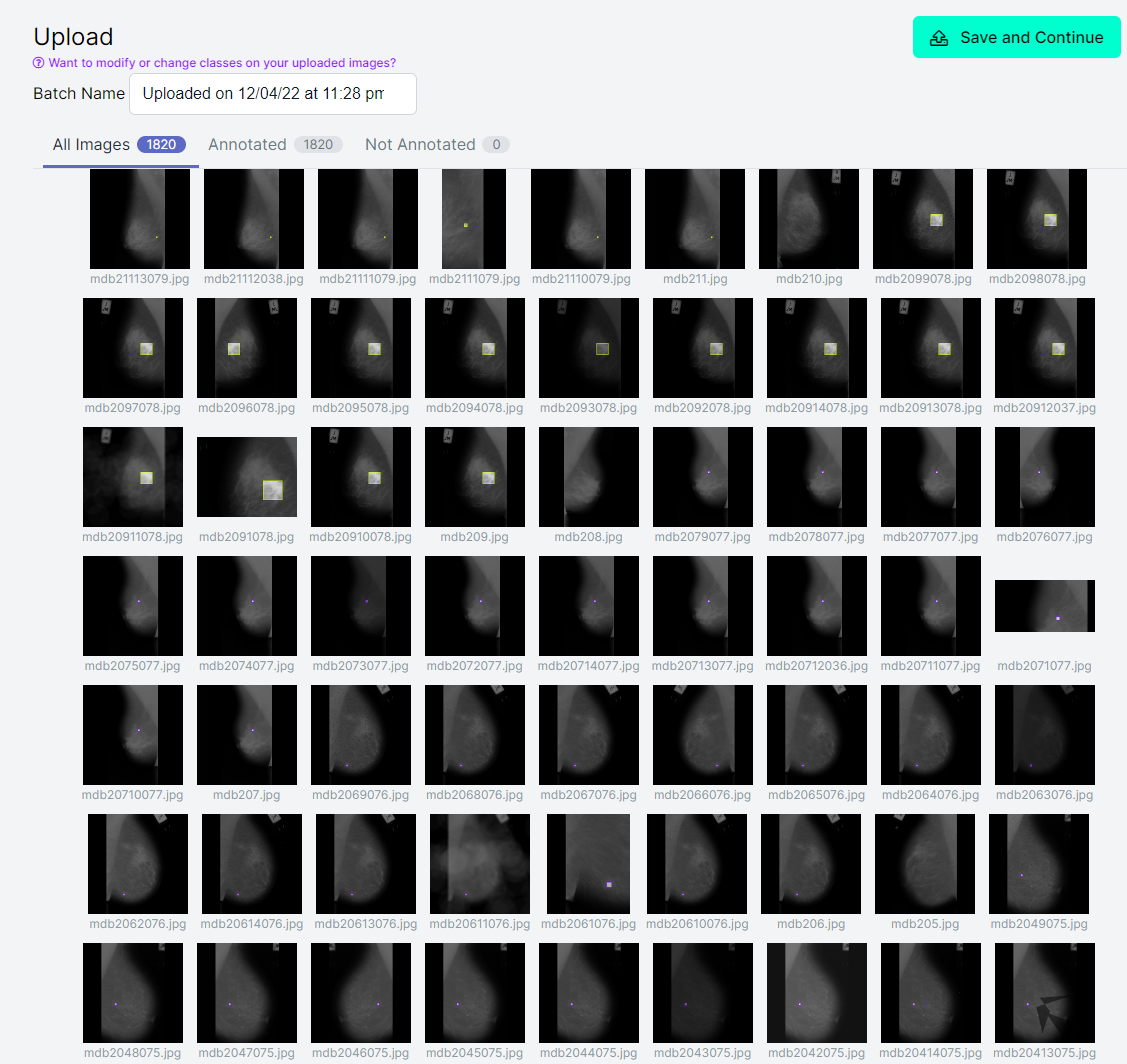


**Augmentation**

For augmentation, we mainly used Roboflow and Albumentation. We used Roboflow to view all the images and the bounding box location. Since our dataset is small, we can check each image and adjust the bounding box if needed. For instance, we removed an image with a bounding box located at an off location. There are also a few images that have bounding boxes including abnormality areas and pure black areas. We edited the bounding box through roboflow.

The majority of the images in our dataset is normal which doesn’t have a bounding box. There are 207 normal images and 119 categorized as abnormal images. We used 9 augmentation techniques including horizontal flip, random brightness, noise, random gamma, blur, jpeg compression, bounding box safe random crop, gaussian noise, sharpen, and the last augmentation combined 4 techniques: horizontal flip, sharpen, random brightness, and blur. This increased the amount of abnormality images from 119 to 1071.

Figure 7: Augmentation from Roboflow

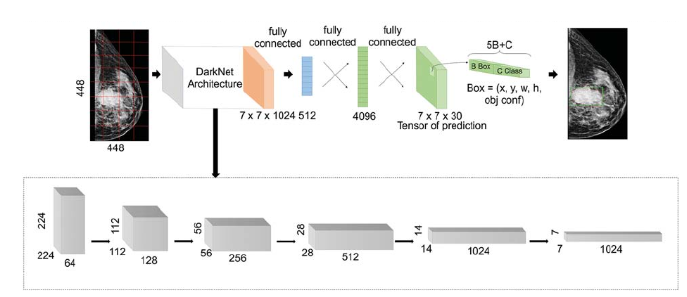


**Methodologies to perform: YOLO**

1. YOLO Definition

You-Only -Look-Once (YOLO) is a fast and low-memory dependence deep learning algorithm to detect objections within one image in real time, which requires only one forward through the neural network to recognize items. The image is split into a certain number of grids that share the same dimension for each title. In this project, we proposed the YOLO model to simultaneously detect and classify breast tumors, including five types of tumor and normal situation. The image is split into a certain number of grids. Then the bounding boxes are added into the image to identify different classes of object. The value of the intersection over union (IOU) determines the final class of the grid, following the combination of the same class of grids.

Figure 6: YOLO Layers



1. YOLO Configuration

Data format

The type of annotation used for YOLO is bounding box, and the YOLO labeling format is used for the YOLO annotation, each image combined with a .txt file with the same file name in the same directory. The annotation file for the corresponding image contains the object class, object coordinates, height and width. The format for the bounding box is set as [class, width, height, bx, by], in which bx and by are the center coordinates of the bounding box.

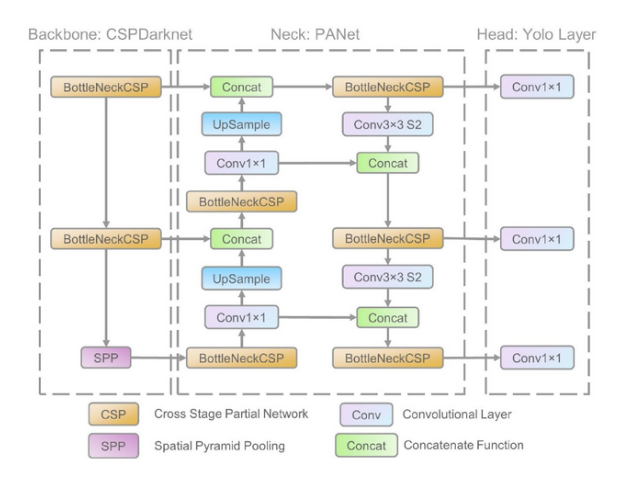
Data splitting

We split our image dataset into 70% training data, 20% validation data and 10% testing data by using Roboflow.

1. YOLO Architecture

YOLO architecture splits the entire image into a certain amount of grids and forward through darknet architecture and fully convolutional layers. There are three components in the YOLOv5 architecture, CSP-darknet53 is used as a backbone to reach deep layers and to overcome the vanishing gradient problem; path aggregation network is a feature pyramid network which is modified as the bottleneck of YOLO, to improve the pixel localization; YOLOv3 or YOLO v4 with three convolutional layers is deployed as the head of the network to predict the locations of bounding boxes, the scores and object classes.

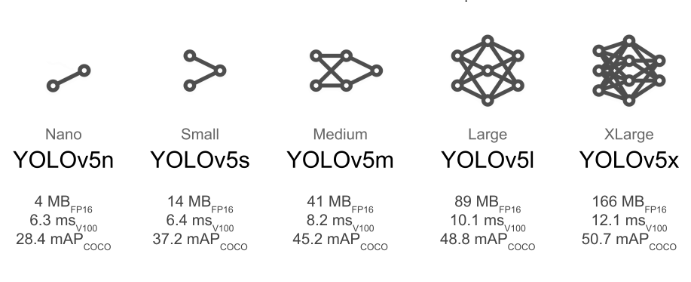
Figure 6: YOLO Architecture



1. YOLO v5 Training

We select two pre-trained YOLO models to train our dataset, one is the YOLOv5s with pre-trained weights from yolov5s.pt and data coco128.yaml, and another is the YOLOv5l with pre-trained weights from yolov51.pt and data coco128.yaml.

Figure 6: Different Sizes of YOLOv5 Models

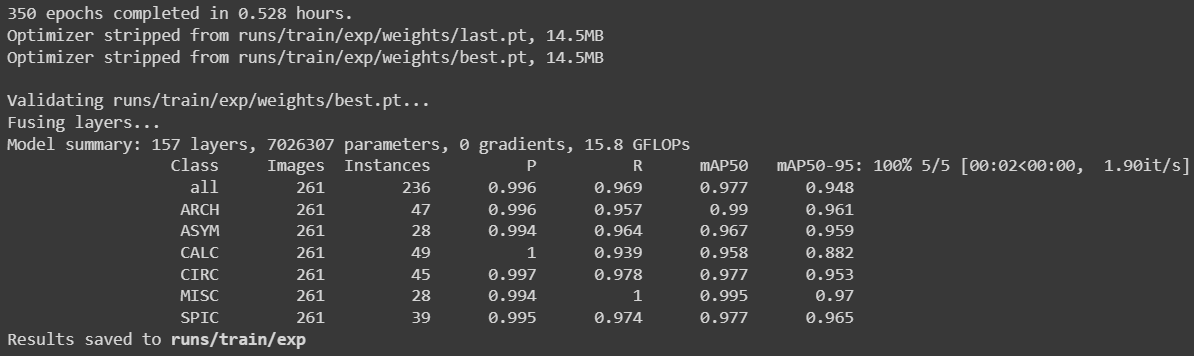


Hyperparameters we selected for YOLOv5l: batch=8, epochs=200, image size=640, lr0=0.01, lrf=0.01, momentum=0.937, weight\_decay=0.0005, warmup\_epochs=3.0, warmup\_momentum=0.8, warmup\_bias\_lr=0.1, box=0.05, cls=0.5, cls\_pw=1.0, obj=1.0, obj\_pw=1.0, iou\_t=0.2, anchor\_t=4.0, fl\_gamma=0.0, hsv\_h=0.015, hsv\_s=0.7, hsv\_v=0.4, degrees=0.0, translate=0.1, scale=0.5, shear=0.0, perspective=0.0, flipud=0.0, fliplr=0.5, mosaic=1.0, mixup=0.0, copy\_paste=0.0

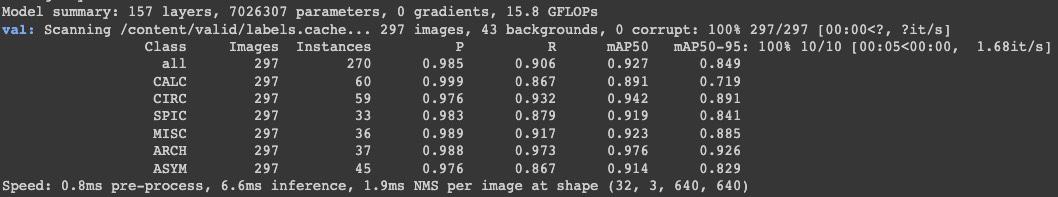
Hyperparameters we selected for YOLOv5s: batch=32, epochs=350, image size=640, lr0=0.01

1. YOLO results

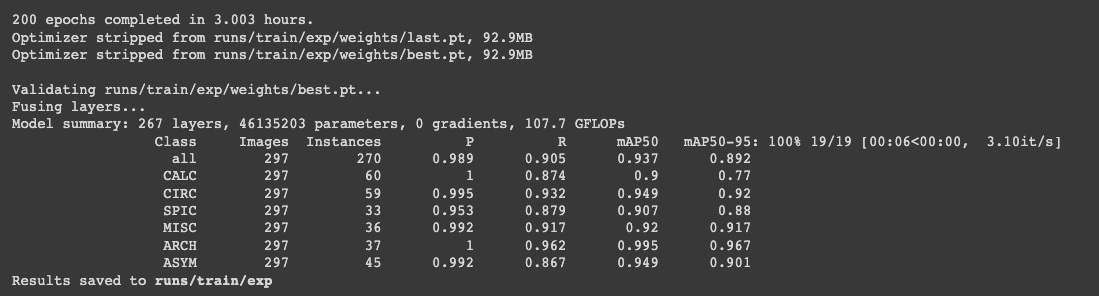
YOLOv5s\_Circumscrbed bounding box



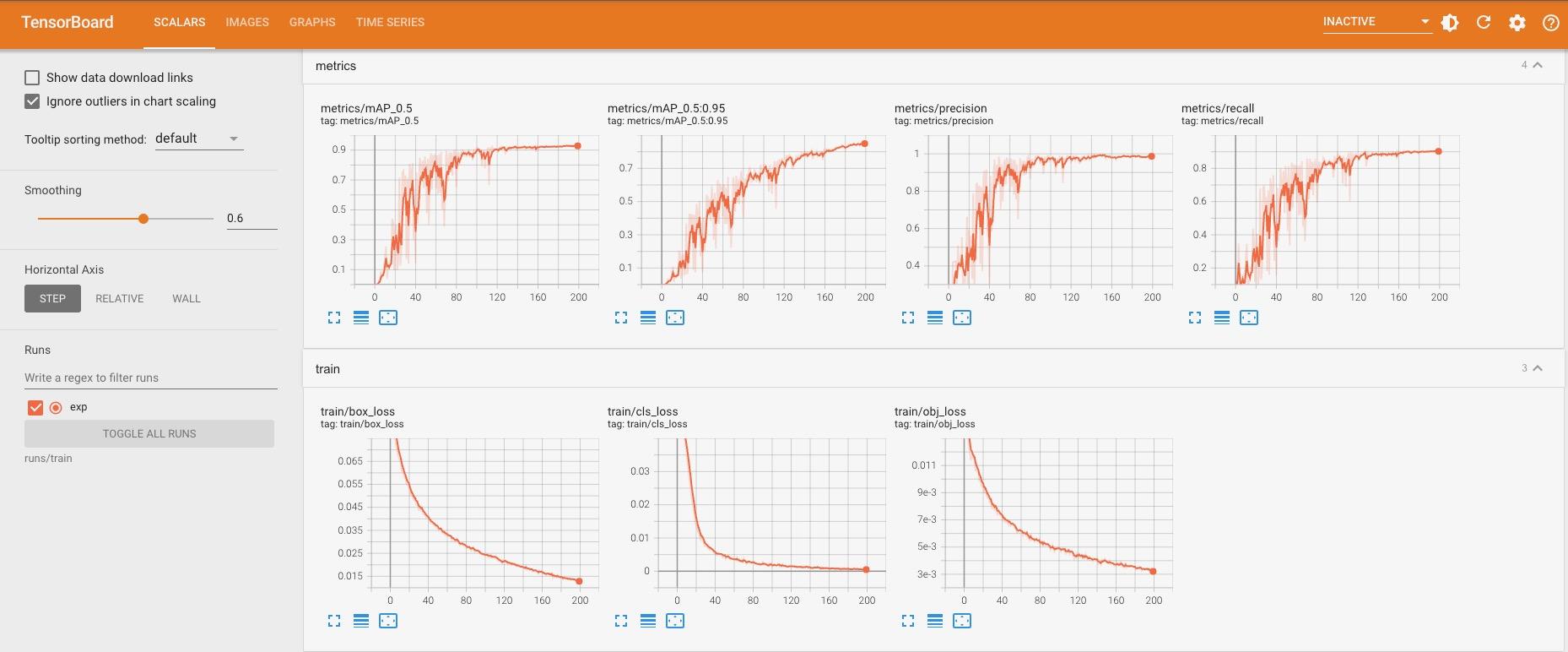
YOLOv5l\_Inscribed bounding box



YOLOv5l\_Inscribed bounding box

****

**YOLOv5s metrics mAP**

****

**Methodologies to perform: RetinaNet**

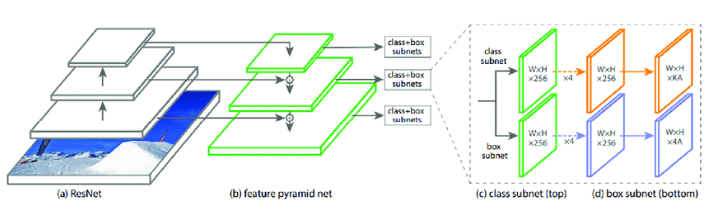
1. RetinaNet Definition

RetinaNet is a single stage object detection model which was presented in a conference by FAIR. The model utilizes Focal Loss which neutralizes the problem of class imbalance which is critical in most of the single stage models. The concept of focal loss involves adding a modulating term to the concept of cross entropy loss in order to concentrate learning on challenging negative examples. It efficiently solves the difficulty of detecting small and dense things. For our purpose we use RetinaNet to predict six different types of abnormalities.

1. Architecture:

It consists of a unified network that is made up of a backbone network and two subnetworks that are specialized for particular tasks. The backbone computes a convolutional feature map that is applied to an input image. The output of the backbone is fed into the first subnet, which then conducts object classification, and the output of the backbone is fed into the second subnet, which then does bounding box regression. The model's backbone consists of a feature pyramid network built on top of the CNN ResNet. The box subnet which is similar to the class subnet is connected to each pyramid level, the only difference being in the linear outputs which happen to be different.

Figure 7: RetinaNet Architecture



1. Configuration:

Data Format:

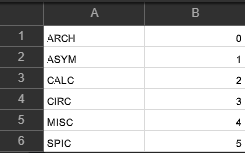
The model is trained on csv files, where the first csv file contains annotations which should have a single annotation per line. Images with multiple bounding boxes are also defined in single line each.

Format : path/to/image.jpg,x1,y1,x2,y2,class\_name

Some of the images which do not contain any bounding boxes are enter with the image but with empty values for the bounding box coordinates also with an empty class name.

The second csv file should have two columns class\_name, id. In our case there are six classes.

Figure 8: Classes of Mias Dataset



Data splitting:

We split our image dataset into 70% training data, 10% validation data and 20% testing data by using Roboflow.

1. Retinanet Training:

The model was trained on pre-trained weights from ResNet50 which is trained on the open image dataset. The model has a default ResNet50 backbone. In clipnorm the gradient is clipped to prevent the parameters from spiking at wrong places.

Epochs: 70

Pre Trained weight: Resnet50\_coco\_best\_v2.1.0.h5

Number of steps: 500

Optimizer: clipnorm 0.001

Even though the model was trained for 70 epochs we observed the least loss at 63 epoch. This saved weights are converted to inference models for validation and testing.

1. Retinanet Results:

Validation Result

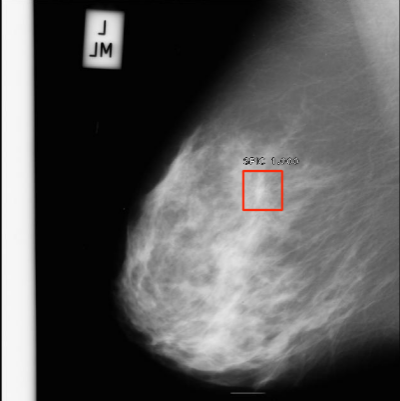
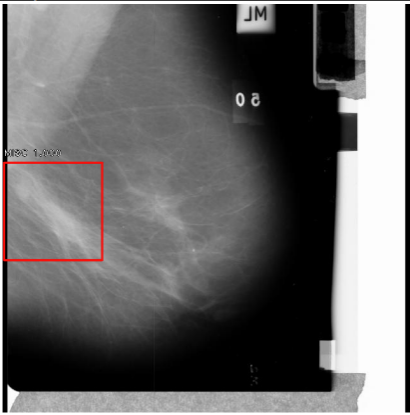
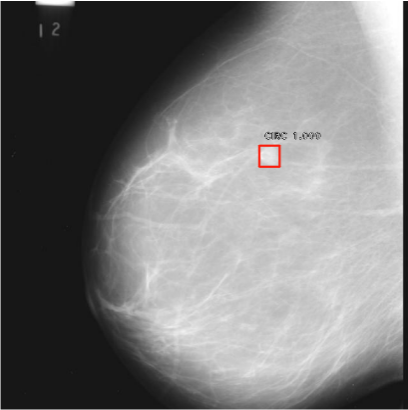
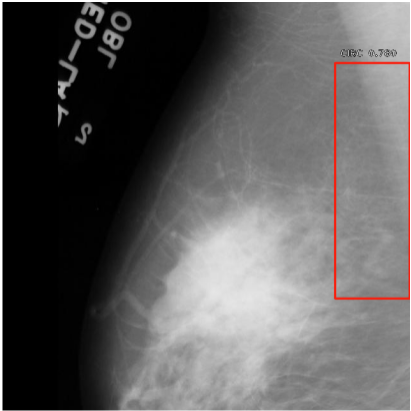
| **Classes** | **Validation - Average Precision** | **Test - Average Precision** |
| --- | --- | --- |
| ARCH | 0.91 | 0.79 |
| ASYM | 0.89 | 0.92 |
| CALC | 0.57 | 0.53 |
| CIRC | 0.95 | 1.0 |
| MISC | 1.0 | 0.94 |
| SPIC | 0.96 | 0.87 |
| **weighted average of precisions** | 0.86 | 0.81 |
| **mAP** | 0.88 | 0.84 |

We see a high accuracy for 5 classes. But the least precision was for one particular class which was 0.53 CALC (calcification). The highest accuracy was obtained for well-defined/circumscribed masses (CIRC).

The average precision for our validation set was 88% and for the test is 84%. The overall accuracy drops due to the CALC class. The reason might be that it is hard to identify.

The inference model shows the following detection

Figure 10: Detection Results



**Evaluation**

|  | Model | Size | Layers | Epochs | mAP50 | mAP50-95 | Train time |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Circumscribed bounding box | YOLOv5s | 640 | 214 | 349 | 0.977 | 0.948 | 0.528 hours (Colab Pro+) |
| Inscribed bounding box | YOLOv5s | 640 | 214 | 200 | 0.927 | 0.849 | 1 hour |
| YOLOv5l | 640 | 368 | 200 | 0.937 | 0.892 | 3.003 hours |
| RetinaNet |  |  |  | 70 | 0.84 |  | 2.48 hours |

**Literature review**

1. SSD: single shot multiple detection

<https://arxiv.org/pdf/1512.02325.pdf>

This paper presents a single deep neural network for multiple category detection. It has extra feature layers to predict the offsets to default boxes after a base convolution layer.The model loss is a weighted add up of localization loss and confidence loss. Its SSD512 model outperforms the Faster R-CNN, and the real time SSD300 model runs faster than the real time YOLO alternative.

1. YOLO Based Breast Masses Detection and Classification in Full-Field Digital Mammograms [<https://www.sciencedirect.com/science/article/abs/pii/S0169260720316564>]

The authors preprocess the mammograms into images for the YOLO model. They use three YOLO models out of which YOLOv3 with anchors is proven for better detection results. ResNet and Inception are used as feature extractors to compare the classification results with YOLO. They conclude augmenting the training is only accurate when applied in realistic scenarios.

1. Chapter 1 - Computer-aided detection of abnormality in mammography using deep object detectors

https://www.sciencedirect.com/science/article/pii/B9780128197400000012?via%3Dihub

This research paper proposed using after pretrained faster R-CNN and YOLO v2 to detect abnormality on a binary image. The authors compare three different backbone architectures: rResNet50, GoogLeNeet, and MobileNet v2 with both object detection models. They evaluate the models with mAP, precision and recall. For YOLO v2, ResNet50 performed the best among the other two CNN architectures. It results in 0.31 mAP. For faster RCNN, it is hard to judge which backbone structure is better since the author wasn’t using the same learning rate or training epochs for faster RCNN. However, GoogLeNet faster RCNN achieved 0.36 mAP, a better mAP in 20 epochs than the YOLO v2 mAP in 100 epochs. In this research, the authors only output the bounding box as their final result, there is no other classification involved.

**Future Work**

* All models can be trained on a bigger dataset, by adding more images.
* The large dataset would enable models to perform much better.
* Additional data augmentation techniques can be tested to get better results.

References:

Jung, H., Kim, B., Lee, I., Yoo, M., Lee, J., Ham, S., Woo, O., & Kang, J. (2018). Detection of masses in mammograms using a one-stage object detector based on a deep convolutional neural network. *PLOS ONE*, *13*(9), e0203355. https://doi.org/10.1371/journal.pone.0203355

https://github.com/fizyr/keras-retinanet