Applied Analytics Practicum

Breast Cancer Detection Project

Team Member Names:

Aditya Thakur, athakur32@gatech.edu
Christopher Figueroa, cfigueroa33@gatech.edu
Kenneth Vallecillo, kgonzalez41@gatech.edu

1 Problem Statement

In medical image analysis, the manual and error-prone task of tumor detection is a significant challenge, despite its role in accurate diagnosis and effective treatment planning [2]. There is a need for the development of advanced, automated tumor detection systems that are capable of accurately localizing tumors with precision while being adaptable to the variability in image quality, acquisition techniques, and patient demographics. This study aims to fill this gap by introducing an object detection model based on innovative regional neural network architectures, targeting an improvement in diagnostic accuracy, and then facilitating other clinical parameter analysis by healthcare professionals. Specifically, this research focuses on enhancing the classification and detection of malign or benign tumors using breast ultrasound imaging, a critical diagnostic tool in regions with scarce healthcare resources.

2 Data Source

The data for this project was sourced from the AI-First Technology medical image repository. In particular, this study contains breast ultrasound images with normal and abnormal cases. The dataset contains the tumor manifestations and their corresponding manually localized bounding boxes for each patient which is key to measuring the model performance in object detection against the ground truth. Additionally, the dataset labels each image as benign or malign, this information will be later used to train the classification model.

Figure 1 and 2 are examples of the images this project utilized. The images provided varied in size from 512x460 to 1256x900 and contained various machine information surrounding the sonogram. The figure also illustrates some technical difficulties. For example, not every image is scanned by the same machine in the same region. Furthermore, ultrasound images are typically blurry and can make it hard for the untrained eye to identify and classify a tumor. Figure 3 shows the annotated red box highlighting the tumor in each of the images above.

Figure 1: Benign Ultrasound Example.

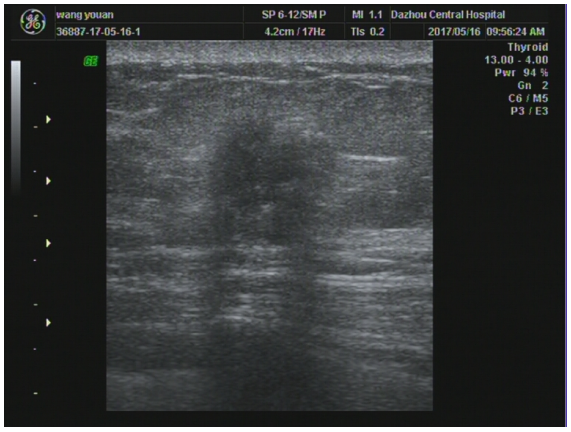

Figure 2: Malignant Ultrasound Example.

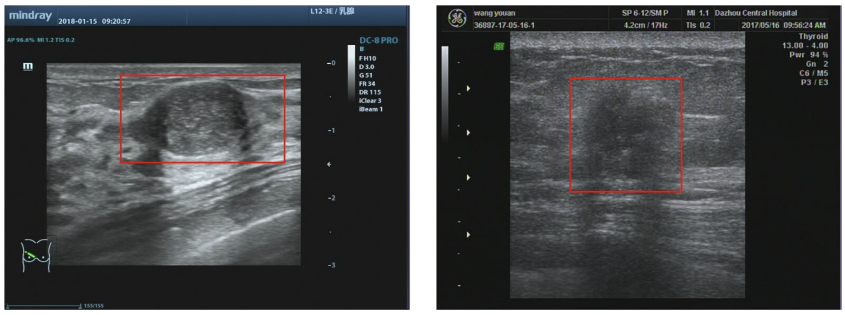

Figure 3: Annotated Ultrasound images.

The given dataset contains over 1,000 images of ultrasound scans, therefore additional augmentations were performed to the images for training. Data augmentation on the images allowed us to grow the dataset to over 9,000 images for training. All images are provided in JPG format with relevant JSON files for annotation. Table 1 shows how many images were used for training and testing.

Table 1: Number of JPG files in different directories.

Type of File	**Number of Files**
Training JPG	1148
Testing JPG	239

3 Methodology

The project utilized convolutional neural networks (CNNs) as the primary methodology for image analysis, as they have demonstrated strong performance in similar tasks, and are considered one of the most successful types of models for image analysis to date [2]. CNNs contain many layers that transform their input with convolution filters [2]. The detection task is to enclose in a box a region of interest (ROI), to achieve a robust object localization we employed the machine learning process illustrated in figure 4, and followed the activities below to implement the given process.

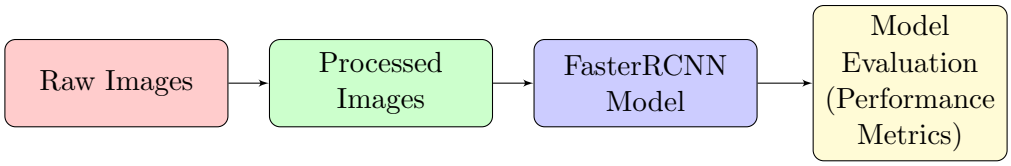

Figure 4: Flowchart of the machine learning process.

1. **Environment and Code**: The project used local and Google Colab environments for Python code execution with GPUs for enhanced computational power, including libraries like Pytorch, as well as tools for data manipulation and visualization. Data storage was managed through Google Drive, accessed via Colab, under a paid subscription due to the demanding nature of the object detection tasks. Development primarily occurred on a local PC to manage resources efficiently by limiting training epochs, and keeping training times reasonable. After developing the model and evaluation functions, training was moved to Google Colab PRO for execution. For replication purposes the source code is available by clicking here code.

2. **Data Preprocessing**: Data preprocessing was crucial for standardizing input image sizes and improving model performance. Images were resized to 600x600 pixels to match the Faster R-CNN input requirements. The size was also selected in order to minimize the potential loss of features that can occur when reducing the size of large images. Normalization was applied to ensure pixel values fall within a range of [0, 255]. Augmentation techniques, including adding noise, blur, improving contrast, and brightness were systematically applied to enhance dataset diversity, aiding the model's generalization ability.

3. **Model Development**: The primary model employed was Faster R-CNN, known for its efficiency and accuracy in object detection and classification tasks. This architecture was chosen for its integrated region proposal network (RPN) and end-to-end training capability. Additionally, we evaluated the Segment Anything Model (SAM) developed by META, which employs a prompt-based segmentation system analogous to language models, offering advanced capabilities in identifying complex object patterns in images. [1].

4. **Model Training and Optimization**: The training of the Faster R-CNN model was monitored using Mean Average Precision (mAP) and Intersection Over Union (IoU) metrics to assess performance. The Adam optimizer was chosen for its effective handling of sparse gradients and adaptability to the model's learning rate. Training involved careful tuning of hyperparameters, including learning rates and batch sizes, to enhance model convergence and accuracy while meeting computing resources requirements. Training also involved the use of different image datasets to determine the effectiveness of various data augmentations.

5. **Future Work and Scalability**: For future work, investigating alternative neural network architectures, such as YOLO or SSD, could offer insights into performance variations across different object detection models. Expanding data augmentation techniques, like advanced geometric transformations and generative adversarial network (GAN)-based augmentations, could further enhance the model's and accuracy. Scalability considerations will focus on optimizing computational efficiency and adapting models to process larger datasets with higher resolution images.

4 Evaluation

In general, the performance of the developed model was evaluated using a separate validation dataset, unseen during training, which was used to assess the model's generalization capabilities. The following evaluation approach was applied.

- **Quantitative Evaluation**: The model was evaluated using common evaluation metrics such as the intersection over union (IoU), Mean Average Precision (MAP), accuracy, sensitivity, or specificity.
- **Qualitative Evaluation**: Visual inspection of the results to ensure the model is identifying the ground truth bounding boxes in the input images, by comparing the predicted bounding boxes to the ground truth so that systematic errors can be identified.

4.1 Metrics

It is important to identify key metrics for adequately evaluating a deep learning model. One of the challenges for the object detection and classification task is the small amount of medical images, which may not provide enough examples to adequately tune all the parameters of a neural network. Therefore, it's important to track various metrics in order to ensure the training set used is adequate in allowing the model to learn good features that can generalize to the test set. The choice of metrics used include the mean average precision (MAP), intersection over union (IOU), sensitivity, specificity, and accuracy.

- **Intersection over Union (IoU)**: This metric is used to evaluate the similarity between two sets, such as the predicted box and ground truth box in object detection tasks. It measures the ratio of the overlapping area between the two sets to their union as illustrated by figure 5. The index ranges from 0 to 1, with 0 indicating no overlap between the sets and 1 indicating a perfect match. The IoU function is defined for two sets A and B as given by equation 1.

$$IoU(A, B) = \frac{A \cap B}{A \cup B}, \leq IoU \leq 1 \tag{1}$$

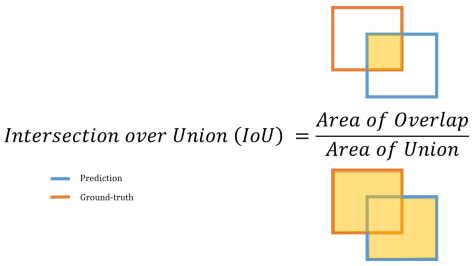

Figure 5: Intersection over Union (IoU).

- **Mean Average Precision (MAP)**: Mean average precision is used to evaluate the performance of object detection models by combining precision and recall. This will give a comprehensive measure of the model's ability to accurately detect objects of interest (ROI) while minimizing false positives. MAP is calculated by first computing the Average Precision

(AP) for each class or object of interest, and then taking the mean of these AP values across all classes. AP is determined by calculating the area under the precision-recall curve for each class, which represents the trade-off between precision (the ratio of true positives to all predicted positives) and recall (the ratio of true positives to all actual positives). MAP is given by the equation 2.

$$\text{MAP} = \frac{\sum_c AP_c}{C} \tag{2}$$

MAP values range from 0 to 1, with higher values indicating better performance in terms of both precision and recall across all classes or objects of interest. A MAP score of 1 implies perfect detection, while a score of 0 indicates poor detection performance.

- **Sensitivity**: Sensitivity or recall measures the ability of the detection model to identify pixels that belong to the bounding box. It is defined as the ratio of true positives to the total number of actual positive pixels or regions. Sensitivity is given by the equation 4.

$$\text{Sensitivity} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \tag{3}$$

True Positives (TP) are the number of correctly identified pixels that belong to the bounding box, and False Negatives (FN) are the number of actual positive pixels that were incorrectly identified as background pixels by the detection model. Sensitivity values range from 0 to 1, with 0 indicating that the model fails to identify any of the pixels of the bounding box and 1 indicating that the model can accurately identify all of the pixels of the bouding vox.

- **Specificity**: Specificity measures the ability of the detection model to identify pixels that do not belong to the bounding box. It is defined as the ratio of true negatives to the total number of actual negative pixels. Specificity is given by the equation 4.

$$\text{Specificity} = \frac{\text{True Negatives}}{\text{True Negatives} + \text{False Positives}} \tag{4}$$

True Negatives (TN) are the number of correctly identified background pixels or regions, and False Positives (FP) are the number of actual negative pixels or regions that were incorrectly identified as part of the bounding by the detection model. Specificity values range from 0 to 1, with 0 indicating that the model may identify many false positive pixels and 1 indicating that the model can accurately identify all the pixels that do not belong to the bounding box.

- **Accuracy**: Accuracy measures the overall ability of the detection model to correctly identify pixels as either belonging or not belonging to the bounding box. It is defined as the ratio of correctly classified pixels to the total number of pixels. Accuracy is given by the equation 5.

$$\text{Accuracy} = \frac{\text{True Positives} + \text{True Negatives}}{\text{True Positives} + \text{True Negatives} + \text{False Positives} + \text{False Negatives}} \tag{5}$$

True Positives (TP) are the number of correctly identified bounding box pixels or regions, True Negatives (TN) are the number of correctly identified background pixels or regions, False Positives (FP) are the number of actual negative pixels or regions that were incorrectly identified as part of the bounding box, and False Negatives (FN) are the number of actual positive pixels or regions that were incorrectly identified as not part of the bounding box by

the detection model. Accuracy values range from 0 to 1, with 0 indicating that the model misclassified all pixels and 1 indicating that the model can accurately identify all the pixels in the image, both in and outside the bounding box.

For object detection tasks, evaluating the accuracy of bounding box predictions requires specific metrics. Intersection over Union (IoU) assesses the overlap between predicted and actual boxes. Precision and Recall address model accuracy in identifying objects and covering all objects, respectively, which is crucial in datasets with class imbalance. Furthermore, Average Precision (AP) per class and mean Average Precision (mAP) across all classes provide insights into model performance on a per-class basis and overall. Together, these metrics IoU, Precision, Recall, and mAP offer a comprehensive evaluation of object detection models, capturing their accuracy and reliability in predicting bounding boxes.

5 Model

5.1 Model Architecture

The architecture employed in this project was the Faster R-CNN, initially introduced by Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun in their landmark 2015 paper, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks" [3]. The Faster R-CNN is a convolutional neural network (CNN) engineered specifically for object detection, representing a significant advancement in the efficiency and accuracy of detecting objects within an image.

The Faster R-CNN architecture integrates a Region Proposal Network (RPN) with a deep CNN, streamlining the process of generating object proposals and then classifying them. The network architecture consists of two main components: the RPN for generating object proposals and the Fast R-CNN detector that uses these proposals to classify and refine their locations. Key elements of the Faster R-CNN architecture include:

- **Region Proposal Network (RPN)**: Generates object proposals by sliding a small network over the feature map obtained from the input image. This network predicts object bounds and scores at each position.
- **RoI Pooling Layer**: Extracts a fixed-size feature vector from each proposal for the detector to process, ensuring that feature vectors are of a consistent size.
- **Fast R-CNN Detector**: Uses the feature vectors from the RoI Pooling layer to classify each proposal into object categories and refine their bounding boxes.
- **Anchor Boxes**: Predefined boxes of various ratios and scales that the RPN uses to adjust the proposals closer to the potential objects.
- **Non-Maximum Suppression (NMS)**: Applied to eliminate redundant overlapping proposals, ensuring that each detected object is represented by a single proposal.
- **Backbone Network**: A pre-trained deep CNN (e.g., VGG16, ResNet) is used to extract features from the input image. The choice of backbone affects the speed and accuracy of the model.
- **Loss Functions**: Comprises a classification loss (to distinguish between object and non-object) and a regression loss (to refine the positions of the bounding boxes). The equation 6

is shown below:

$$L(\{p_i\},\{t_i\},\{v_i\}) = \frac{1}{N_{\text{cls}}}\sum_i L_{\text{cls}}(p_i, p_i^*) + \lambda\frac{1}{N_{\text{reg}}}\sum_i p_i^* L_{\text{reg}}(t_i, v_i) \tag{6}$$

A visual representation of the Faster R-CNN architecture is shown in figure 6, illustrating the flow from input image through the backbone network, RPN, and Fast R-CNN detector, culminating in the output of detected objects with their classifications and bounding boxes.

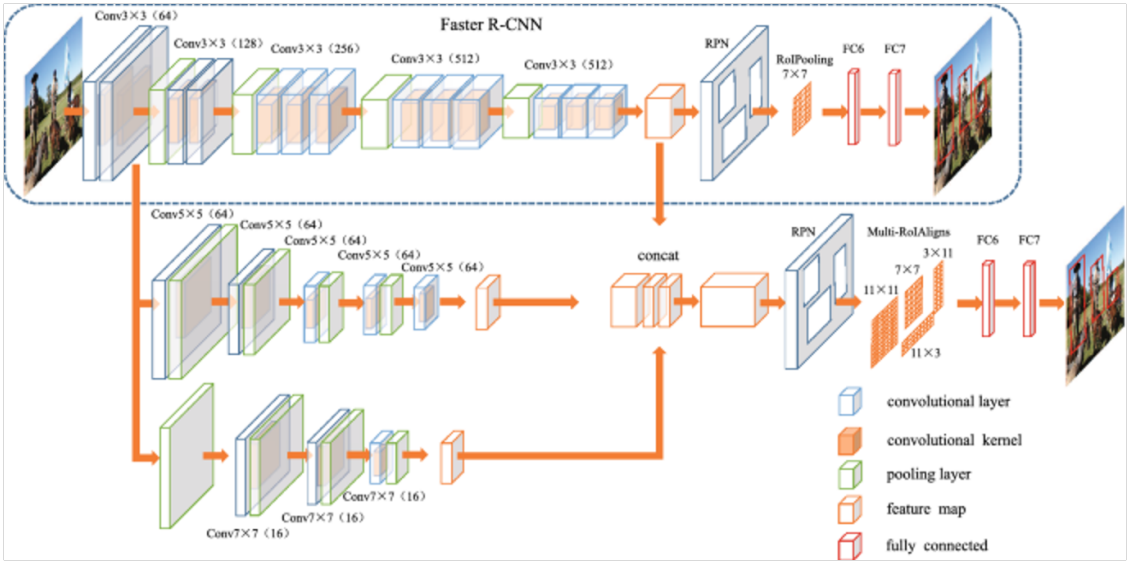

Figure 6: Faster R-CNN architecture.

The Faster R-CNN model operates as follows: Let $x \in \mathbb{R}^{H\times W\times C}$ denote the input image with height H, width W, and channels C. The process begins with the input image passing through the backbone network to produce a convolutional feature map. The RPN then scans this feature map using anchor boxes to generate proposals, which are refined and classified by the Fast R-CNN detector. The entire process is optimized end-to-end with a combined loss function addressing both the RPN and Fast R-CNN components.

$$RPN = \text{GenerateProposals}(FeatureMap) \tag{7}$$

$$Proposals = \text{Refine}(RPN, AnchorBoxes) \tag{8}$$

$$DetectedObjects = \text{FastRCNNDetector}(Proposals, FeatureMap) \tag{9}$$

This architecture effectively marries the proposal generation and object detection tasks into a single, cohesive model, enhancing the speed and accuracy of object detection workflows.

5.2 Training Procedure

The Faster R-CNN model was tailored for a 600x600 input size and trained over 10 epochs using the Stochastic Gradient Descent (SGD) optimizer, with parameters set to a learning rate of 0.005, momentum of 0.9, and weight decay of 0.0005. The Faster R-CNN model uses the multi-task loss

function, which combines classification and regression losses, for backward gradient calculations. The batch size was set at 10 to optimize computational efficiency and learning accuracy while also being capable of training on GPUs. Model performance was evaluated using Mean Average Precision (mAP) and Intersection over Union (IoU) metrics, providing a holistic view of its accuracy in tumor detection and classification in ultrasound images. Detailed training metrics, including loss and mAP, were systematically recorded for each epoch, enabling thorough post-training analysis. The Segment Anything Model (SAM) was utilized in its pre-trained state, bypassing additional training for this project.

5.3 Data Augmentation

For the Faster R-CNN model training, data augmentation played a crucial role in enhancing the robustness of the model. The images underwent a series of transformations to introduce variability and mimic different scanning conditions. The processed images were resized to 600x600 pixels as part of the standardization, maintaining the aspect ratio and ensuring no loss of information. The augmentation techniques included noise additions using Gaussian noise, salt-and-pepper noise, and contrast-limited adaptive histogram equalization (CLAHE). At least one of these augmentations was applied to the training images, ensuring that the model encountered diverse variations of the data during each epoch, thus preventing overfitting and improving the model's ability to generalize to new, unseen data. The precise augmentation techniques and parameters were carefully chosen to best simulate the variations observed in clinical ultrasound imaging, enhancing the model's learning and detection capabilities.

5.4 Results

Following data augmentation and the training of a basic model, multiple experiments were run to fine-tune the performance of the model. The experiments included manipulating different augmentation techniques across multiple epochs. Validation metrics were acquired by resizing the images to 600 by 600 during the training process. However, when the final model was acquired after training, bounding boxes and predictions were made on the original validation images. During final predictions, it was decided to use a green bounding box to detect a tumor that is predicted to be benign and a blue bounding box to detect a tumor that is predicted to be malignant.

The analysis in 7 shows a more robust model training with 10 epochs, it shows how when epochs increase the model starts improving its capability to delineate the structures closer to the ground truth.

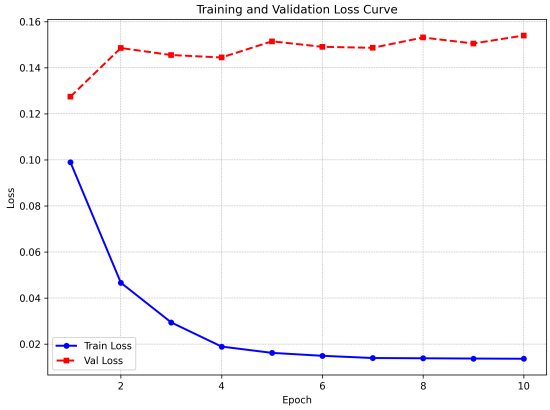

Figure 7: Training and Validation Loss Curves

Another aspect to observe is any potential overfitting of the model, if the curves increase after a plateau, the model overfits the data and loses its generalization capabilities for new images. In this case, the model with 10 epochs plateaus at a satisfactory value close to 0.9 and does not increase after which indicates a good bias-variance tradeoff. Furthermore, the analysis in 8 shows a higher Mean Average Precision (MAP) when a model has more epochs. Similar to the loss curves, we see that 10 epochs show the most robust training model.

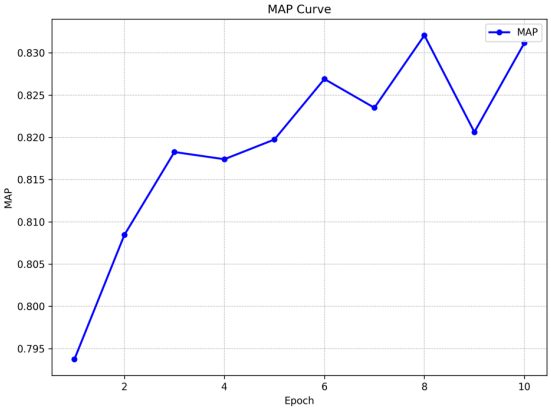

Figure 8: Mean Average Precision Curves by number of Epochs

Table 2 presents the performance metrics for the RCNN-tuned model at 10 epochs. These metrics show that the Faster R-CNN model is a strong model for tumor segmentation and classification, achieving a Mean Average Precision (MAP) of **0.7972**, mean Intersection over Union (IoU) of **0.7029**, a sensitivity of **0.9695**, a specificity of **0.8881**, and accuracy of **0.9156**.

While it was considered to train the model for more than 10 epochs, it was found that performance did not improve significantly after 10 epochs. After 10 epochs, the training and validation losses flattened out and did not change significantly, showing that the model was not learning any new features or improving on its ability to generalize on the validation set. It was decided that ten epochs was satisfactory as less training time was needed for 10 epochs.

The precision of our model across different IoU thresholds is depicted in Figure 9. The Mean Average Precision (MAP) typically decreases as the IoU threshold increases, which is expected as a higher IoU threshold demands a stricter overlap between the predicted and ground truth

Table 2: Summary of Object Detection Model Performance Metrics.

Model	**Avg. IoU**	**MAP**	**Sensitivity**	**Specificity**	**Accuracy**
FRCNN Model	0.7029	0.7972	0.9695	0.8881	0.9156

bounding boxes to be considered correct. The graph shows the trade-off between precision and IoU threshold, highlighting the importance of choosing an appropriate IoU threshold for validating model performance.

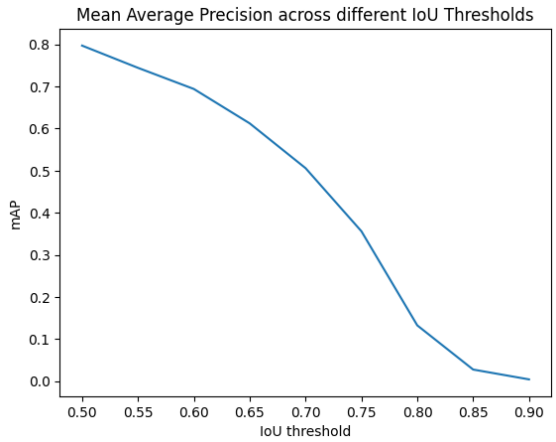

Figure 9: Mean Average Precision by IoU Thresholds.

The Faster R-CNN model had a very good average IoU across the validation set. There were only two images where the model was not able to detect a tumor. From the images where the model detected a tumor, the model had an accuracy of **91.56** percent. The only concerning metric comes from the confusion matrix shown in 10, where the model incorrectly classified 16 malignant tumors as benign. This corresponds to a false negative rate of **6.75** percent. This shows that the model needs some further improvement in being able to distinguish malignant tumors. While the model exhibits a small false positive rate, only classifying 4 benign tumors as malignant, this type of error would be more tolerable than missing a developing cancer.

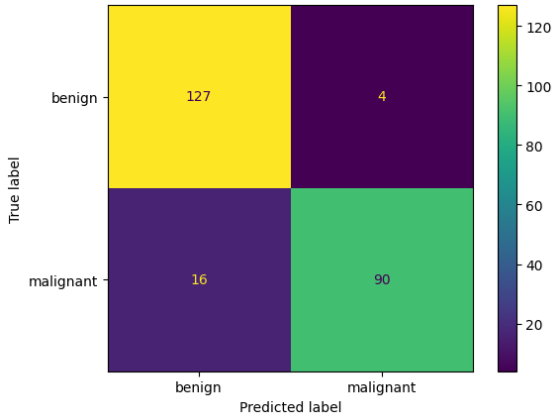

Figure 10: Confusion Matrix of Model using the Validation Set.

An interesting observation during training experiments was that training the model using rotated images resulted in a worse-performing model. When the model was trained using rotated images, it was found to identify all of the test images as malignant and did not properly detect the tumor. It was decided to simply train a model using only noise augmentations as opposed to rotations. One possible reason for this is that a rotated image teaches the model bad features with respect to identifying a tumor. The breast tissue images are taken such that the tissue is oriented roughly horizontally. It is possible that rotating the tumor teaches the model to look for tumors in a vertical orientation, which is not found in any of the validation images. While image augmentations can help develop a more robust model, it appears that rotating the medical images hurt model performance rather than help. Figure 11 shows examples of images created during the training process where model training performance varied widely.

It was also discovered during training and testing of the model that image cleaning using SAM was not entirely necessary. There were various issues when using SAM to clean the images, which included poorly sized segments and the loss of features from the medical image. During the development of the model, it was decided to simply use and perform various augmentations on the original images. It appears that as the model trains over more epochs, it learns to ignore the surrounding medical machine information and focus on the medical image that we were originally trying to crop out. This convenient feature learning from the model reduced the need for image cleaning and made the training process more efficient.

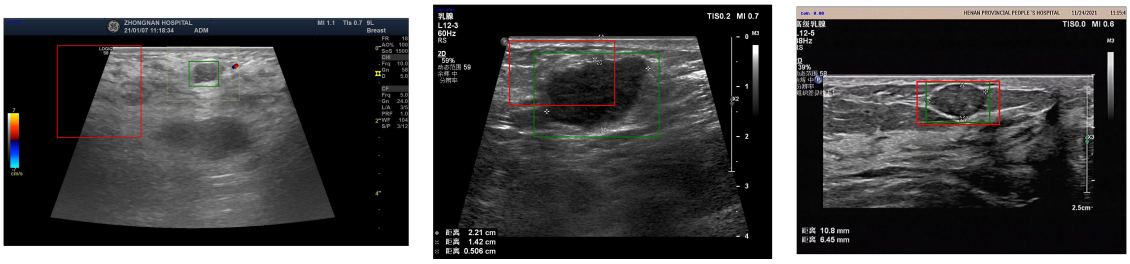

Figure 11: Good, Acceptable, and Bad Predictions

Figures 12 presents examples of the final model predictions with the original Ultrasound images with an overlay of the predicted tumor location and the manually identified ground truth bounding box in red and the predicted box in light blue and green.

6 Conclusions

This study demonstrated the application of the Faster R-CNN model to effectively detect and classify breast tumors in ultrasound images, showcasing the potential of advanced deep learning techniques in medical imaging analysis. Through meticulous data preprocessing and augmentation, the model was trained to recognize and localize tumors with high accuracy, as evidenced by the performance metrics.

The integration of Mean Average Precision (MAP) and Intersection over Union (IoU) as evaluation metrics provided a comprehensive assessment of the model's accuracy in both detecting the presence of tumors and precisely delineating their boundaries. The results, 0.7972 mAP, and 0.7029 IoU, indicate a strong capability of the model to contribute significantly to the diagnosis and treatment planning in oncology.

While the SAM model offered the potential of quick cleaning, the model demonstrated the ability to ignore unnecessary information and focus on the area of interest. The capability of the

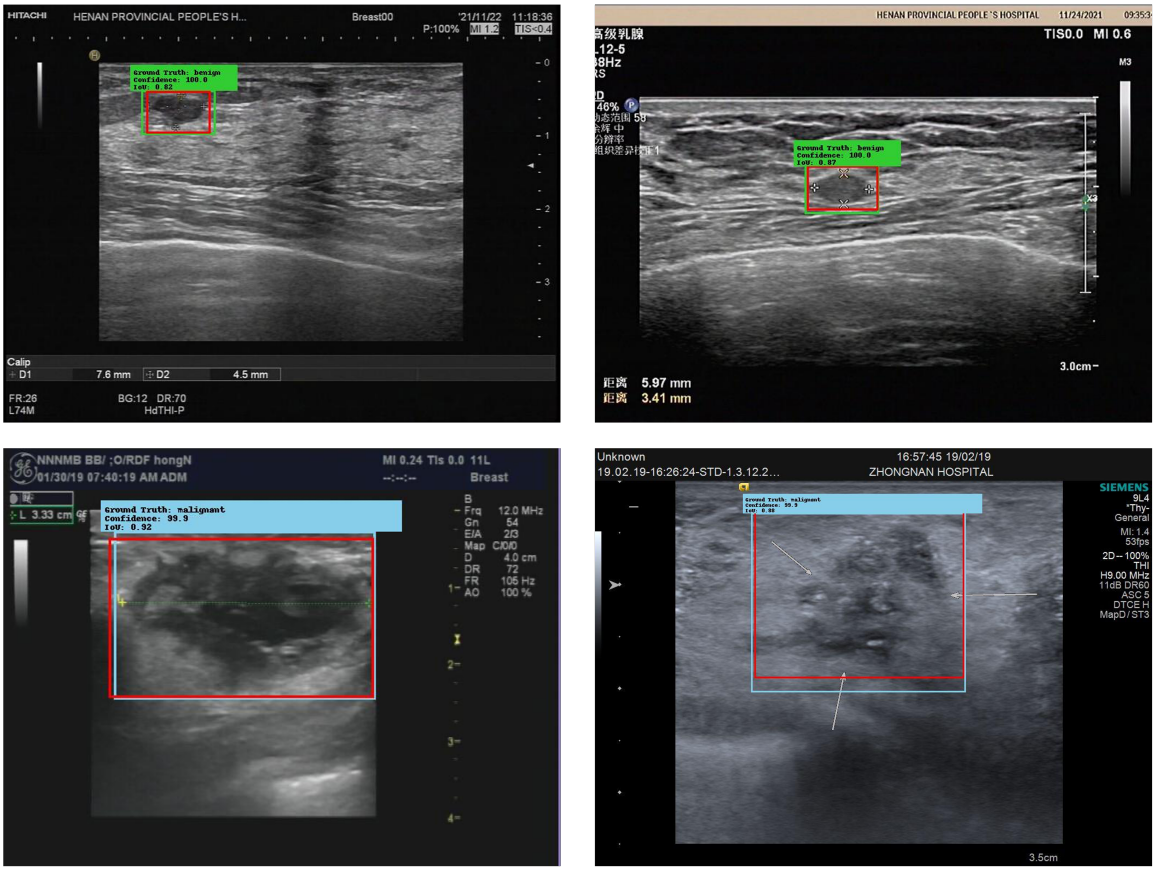

Figure 12: Final Model Predictions

Faster R-CNN model removed the need for image cleaning and conveniently allowed for more focus on training and developing augmentation techniques to improve model accuracy. Further work can expand on other noise-addition techniques and other methods to develop model robustness. It is also desirable to determine methods to reduce the false-negative rate, potentially using adversarial training approaches to improve malignant tumor detection.

Overall, the project shows the strong potential of using a Faster R-CNN network. A model was developed and managed to acquire over 90 percent accuracy using a relatively small image dataset. Augmentations not only allow for growing the dataset but also allow the model to learn robust features that allow for good accuracy. The model can only get better as more medical images are provided. For further information refer to the following notebook. Colab Notebook.

References

[1] Alexander Kirillov et al. "Segment Anything". In: *arXiv:2304.02643* (2023).

[2] Geert Litjens et al. "A survey on deep learning in medical image analysis". In: *Medical Image Analysis* 42 (2017), pp. 60–88. URL: https://doi.org/10.1016/j.media.2017.07.005.

[3] Shaoqing Ren et al. "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.6 (2017), pp. 1137–1149. URL: https://doi.org/10.1109/TPAMI.2016.2577031.