Project Summary- Great Barrier Reef COTS Detection

Aashika Padmanabhan, Nikita Pardeshi, Mitali Shah, Shazia Sulaiman, Rae Zhang

Introduction [1,7,13]

The Great Barrier Reef is the world's largest coral reef system. Coral reefs are home to over 7000 species and providers of important services including fisheries, medicines, tourism etc. Coral reefs are in great danger with the planet having lost 50% of it. Coral predation by Crown-of-thorns-starfish (COTS) accounted for 42% coral loss [4]. Hence, detection and control of COTS will have a huge impact on coral reef protection. Through this project we intend to use the images collected by The Great Barrier Reef foundation and detect the presence of COTS using State of Art Neural Network models. This is one of the Kaggle competitions and we have extracted the data from Kaggle which mainly contained all the image frames from three videos and train and test csv files.

Models Used

We have finalized on 2 models - Faster RCNN and YOLOv5 and below sections will provide more details on each of the models.

Faster RCNN [2,3,8]

Due to its speed, we decided to explore the Fast-RCNN model. This is an advancement over RCNN because instead of dividing an image to 2000 regions the image is fed to one ConvNet to create feature maps and these reduced feature maps are fed to a network called Region Proposal network (RPN) which produces the Regions of Interest (ROI) [3]. The RPN layer makes the FasterRCNN model faster than other RCNN models. The region proposals are reshaped with a ROI pooling layer which is then used for image classification and predicting the bounding boxes values.

Data Processing for Faster RCNN [5]:

The following steps were performed:

1. On the training data , the number of bounding boxes for each image data were calculated using annotation labels.
2. Data was filtered by removing rows with 0 bounding boxes.
3. Labels (annotations) were also mapped to .txt files and saved in separate folders.
4. New column - Box location label was added as a list of list by consolidating all bounding boxes within an image

Training and Testing:

A subset of data was selected as the model is notoriously slow and was split into 80-20 train and test dataset to 3 epochs. As the model was implemented using Pytorch this is converted to a Data Loader.

Model Details:

- Model used - Pretrained version of fasterrnn_resnet50_fpn. The pretrained ResNet-50 is a CNN model that is 50 layer deep.We loaded the pretrained version and trained on the images from on the COTS dataset.
- Optimiser - Stochastic Gradient Descent
- Activation function - ReLu

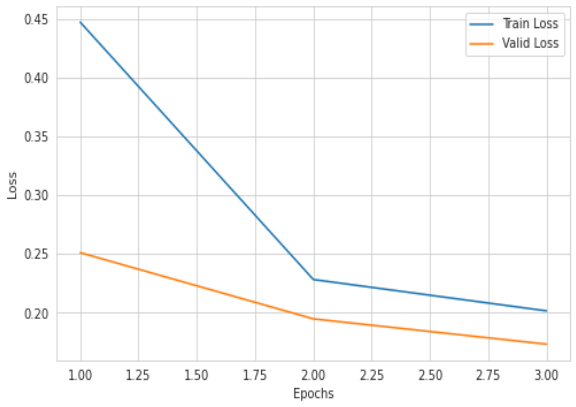

Figure 1: Training and Validation loss

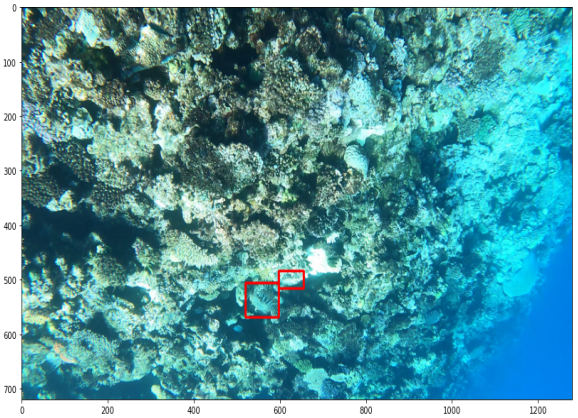

Figure 2: Samples of COTS detected using Faster RCNN

From Figure 1, we can see that the loss is reduced for both Training and Validation dataset. Figure 2 is an image with COTS detected using the fasterrnn_resnet50_fpn model.

YOLOv5

Given its speed and precision, YOLO is one of the most well-known object detection techniques. Here we have used YOLOv5s [6].

Data Preprocessing for YOLOv5 [12]:

The following steps were performed to get the data in appropriate format:

1. Using the training data, annotations (x, y, width, height) were formed and the number of bounding boxes were calculated. Bounding boxes were defined. YOLO expects data in a particular format as shown below: (x_centre, y_centre, box_width, box_height)
2. Using image_ids, the image frames were mapped with their respective image paths from different videos and appended to the train dataset.
3. Data was split into train, test and validation datasets with the following sizes:
   Train: (2951, 9), Test: (984, 9), Validation: (984, 9) and the corresponding images are mapped. Labels (annotations) were also mapped to .txt files and saved in separate folders.

Training and Testing [7,8,9,12]: (YOLOv5 package was downloaded from https://github.com/ultralytics/yolov5.git)

1. Training was performed on all the 2951 images with image size = 1280, batch size = 16 and 30 epochs on google colab with GPU accelerator. Extra computing units were purchased on google colab for training. The notebook can be accessed **here**. Validation was done on the 984 images.
2. Using the above best weights (saved in best_weights.pt file), the model was tested using the predefined detect.py code from YOLOv5 and the following results were obtained:

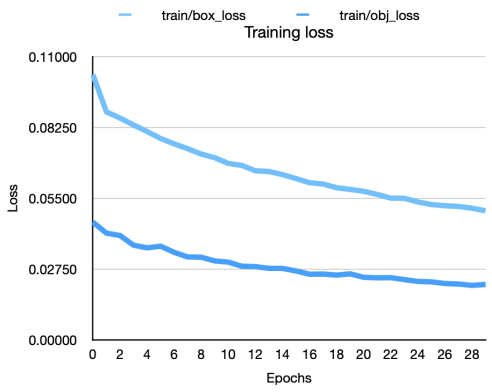

Figure 3: Training Loss

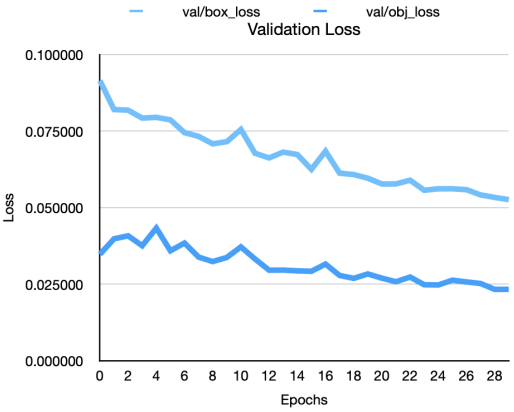

Figure 4: Validation Loss

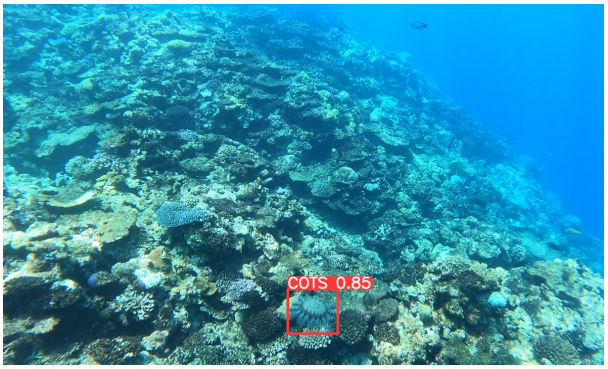

Figure 5 & 6: Samples of COTS detected using YOLOv5

The training and validation loss for detecting bounding boxes and objects decreased with the increase in number of epochs and the best epoch was 29 (starting from 0). From the detected images we can see that the model does a good job, if not excellent, in detecting the COTS. After hyperparameter tuning of trying different confidence levels, we decided to use 0.60 confidence level for our COTS detection model.

Conclusion [10]

It was found that the Faster RCNN model took a very long time to run and the accuracy of the model was not as comparable as that of the YOLOv5s model.

Future Scope[12]

Ensemble models can be applied to the dataset and more epochs can be run to increase the accuracy. Newer versions of the Yolo model can also be experimented.

Relevant Folders

Due to the vast size of the models, dataset etc., we have put all the relevant files under organized folders as below.

1. Drive folder with all YOLOv5 relevant files: https://drive.google.com/drive/folders/1bzg25F-sFlvD00n48j-YW_pRadOfYArk?usp=share_link
2. Drive folder with all Faster -RCNN relevant files: https://drive.google.com/drive/u/1/folders/1WPk0_J3Ptce07a7UTxrVOT9svdRE_HgF

References

1. Crown-of-thorns starfish control. Great Barrier Reef Foundation. (n.d.). Retrieved December 6, 2022, from https://www.barrierreef.org/what-we-do/reef-trust-partnership/crown-of-thorns-starfish-control
2. Gandhi, R. (2018, December 3). R-CNN, Fast R-CNN, Faster R-CNN, YOLO — Object Detection Algorithms. Medium. https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e
3. DeepLearning. (2020, December 4). R-CNN, Fast R-CNN and Faster R-CNN explained. YouTube. https://www.youtube.com/watch?v=vr5rs_cTKCs
4. Crown-of-thorns starfish - reef resilience. (n.d.). Retrieved December 6, 2022, from https://www.reefresilience.org/pdf/COTS_Nov2003.pdf
5. you359. (n.d.). You359/Keras-FASTERRCNN: Keras implementation of faster R-CNN. GitHub. Retrieved December 6, 2022, from https://github.com/you359/Keras-FasterRCNN
6. Solawetz, J. (2022, November 21). What is YOLOv5? A Guide for Beginners. Roboflow Blog. https://blog.roboflow.com/yolov5-improvements-and-evaluation/
7. Garg, A. (2021, December 14). How to Use Yolo v5 Object Detection Algorithm for Custom Object Detection. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2021/12/how-to-use-yolo-v5-object-detection-algorithm-for-custom-object-detection-an-example-use-case/
8. Sharma, P. (2020, May 24). A Step-by-Step Introduction to the Basic Object Detection Algorithms (Part 1). Analytics Vidhya. https://www.analyticsvidhya.com/blog/2018/10/a-step-by-step-introduction-to-the-basic-object-detection-algorithms-part-1/
9. Tips for Best Training Results 📌 - YOLOv5 Documentation. (n.d.). https://docs.ultralytics.com/tutorials/training-tips-best-results/
10. Mean Average Precision (mAP) in Object Detection. Kukil. https://learnopencv.com/mean-average-precision-map-object-detection-model-evaluation-metric/
11. Bochkovskiy, A. (2020, April 23). YOLOv4: Optimal Speed and Accuracy of Object Detection. arXiv.org. https://arxiv.org/abs/2004.10934
12. Maunil2k/The-Great-Barrier-Reef-Kaggle: YOLOv5 trained on custom dataset for COTS detection - Kaggle competition. Maunil2k. https://github.com/Maunil2k/The-Great-Barrier-Reef-Kaggle
13. TensorFlow - Help Protect the Great Barrier Reef | Kaggle. (n.d.). https://www.kaggle.com/competitions/tensorflow-great-barrier-reef