University of Essex
Department of Computer Science and Electronic Engineering

MSc Dissertation

Automatic Pixelwise Annotation Of Coral Reef Substrates Using Deep Convolutional Networks

Renjith Nataraja Pillai
Student ID: 2111151

Supervisor: **Dr Jon Chamberlain**

December 14, 2022
Colchester

Contents

List of Figures

List of Tables

Abstract

This paper propose a deep-convolutional network model to segment coral reef images into different types of substrates.The relevance of the project is discussed in the paper, followed by a detailed description of the methods and techniques used for data pre-processing, model architecture and testing.The project can be divided into two parts, the first one is the annotation of the substrates in the given train images and the second one is the segmentation task.The performance of the model is evaluated using a metric called Intersection over union and the objective was to obtain a value above 0.5 which is considered as a standard value for normal segmentation tasks but here, the task is more complicated due to the large number of classes and quality of the images. The results got for the base line model was (sample set 1: dominica-cabrits : 0.152), but after using different image enhancement methods and with optimum parameter values the performance of the model is increased near to good performance (sample set 1: dominica-cabrits : 0.458). The list of possible approaches/Methods that can be applied for future studies were also listed at the end.

Keywords: semantic segmentation, coral reef segmentation, automatic segmentation

CHAPTER 1

Introduction

The underwater structures known as coral reefs are made of the coral or polyp skeletons, which are colonies of marine colonial invertebrates. The delicate coral bodies were protected by these hard exoskeletons which are formed by extracting calcium carbonate from sea water. They grew in the exoskeleton of their ancestors and when they die, they leave their skeleton for the formation of new polyps. This process continues and will develop into a massive structure of corals. They can be found in almost all part of the sea irrespective of the surroundings, since they have a very good adaptive capability with the environment they grow [1]. There are different species of coral reefs found on earth, due to the diversity of this they are often called as rain forest of the sea. Depends on the place where they grow, they exhibit different characteristics and are classified mainly into two: soft and hard coral. The classification is made based on the number of tentacles that each polyp have. Soft corals have eight tentacles and they mostly look like colourful plants were as the hard corals have multiples of six tentacles. There are many benefits of coral reefs, like maintaining an ecosystem around the environment where they grow by providing shelter and food to fish and other sea organisms. The Northwest Hawaiian Island coral reefs is a best example for this. There are around 7000 species of fish, plants, and other organisms were found in this area. They also support local economies, provide opportunities for recreation, and act as a barrier against erosion and storm damage. Over 500 million people rely on reefs for their shelter, livelihood, and food. Fishing, diving, and snorkelling on and near reefs bring in millions of dollars for neighbouring businesses [2].

In these days the coral reefs are in severe threats. The natural calamities like storm, predators, and diseases are affecting the existence of coral reefs and the unscientific fishing practices and pollution caused by humans are also leading to coral bleaching and destruction. Rigorous restrictions should be put in place, and continual monitoring should be done, to protect these coral ecosystems. Due to the numerous species that make up coral reefs, finding them and monitoring their health are challenging tasks. The size, shapes, and colours of these creatures vary from species to species, also its challenging for people to enter places where they are found. Now a days, the modern technologies like RC drones, under water cameras etc can assist humans in these difficult tasks. Its a difficult task to consolidate and analyse the data generated by these technologies. The deep learning methods can be used for this. These algorithms will simplify the process for humans, from detecting the presence of coral reefs to complicated annotation and segmentation of various coral reefs. Even while it makes the process simpler, using these strategies in real-world situations presents many difficulties. Here in this study, we are interested in automatically identifying regions of interest and labelling them correctly for monitoring coral reefs. Advances in autonomously annotating photos for complexity and benthic composition have been encouraging. Within the next 30 years, it's possible that coral reefs and the ecosystems they sustain could disappear. This catastrophe, which will also result in a global humanitarian crisis for the billions of people who depend on reef services, will lead to the extinction of several marine species. By monitoring the evolution and composition of coral reefs, we can help to prioritise conservation efforts [3].

Objective

The objective of the project is to create a semantic segmentation model that can classify the different coral substrates present in the images that taken using the under water cameras at different locations. The project can be divided into two parts, the first one is the annotation of the substrates in the given train images and the second one is the segmentation task. The project is done as a part of ImageCLEF coral task which started in 2019. After the segmentation task the model is evaluated using the popular evaluation methods and the results can be comparable with the previous studies. It should note that due to the system limitations the model is created based on a sample dataset and the these datasets are generated based

on location criteria. The model performance on all these datasets are listed out and done a detailed analysis in the discussion part. Also different parameter settings were used in the model to get the best results and different image enhancement techniques like augmentation, colour filters were applied on the dataset to identify their effects on the model performance. The detailed analysis along with the images of the prediction is also included in the paper for better understanding.

The paper is written in 5 sections, in the first section a brief introduction of coral reefs are given along with a detailed explanation of the importance of this study in the current scenario. Followed by the introduction, a comprehensive study of 3 different papers from the previous editions were given in the literature survey. We chose the papers based on the model performance and results they obtained during the competition, also it is noticeable that all these papers follow different methods and the conclusions and takeaways from each paper is noted at the end. The next section is methodology where we explain the methods we followed in this project in detail. Since the whole project can be divided into two task the methodology section is also divided in to two and the step-by-step procedure of doing each task and the evaluating methods were presented with relevant tables and images. The fourth section is the results, were we discuss about the results that we got with different datasets and different model parameters. For the convenience of the reader the results are also visually represented along with numerical values. The final section is the conclusions and suggestions, here the final conclusions that we derived from our project and possibilities of future studies were discussed.

CHAPTER 2

Literature Survey

The Coral Reef Image annotation and classification task is a part of ImageCLEF which is launched in 2003. The first edition of the ImageCLEF coral was in 2019, second and third edition was in 2020 and 2021 respectively. Like we discussed in the previous section the objective of the task is to classify the different coral substrates from the under-water images. Due to the quality of the images, the conditions in which they were taken, and 13 class counts increases the complexity of the task. Even though, over these years there was a significant improvement in the model performance and got some promising results from the participants. Also, more images are added to the dataset every year which may also helps to get a better task performance. In the third edition a complete set of images were available to create the 3D reconstruction of the marine environment and studies were done whether the cross learning can increase the model performance. But adding more images from different locations and varied morphology increased the complexity of the task and hence the results were poor compared to the previous editions. This section will compare and analyse three different papers that got good results in the previous editions and will discuss about their methods, the model used, the challenges they faced and the how these studies will help us to create a model which will give a better performance.

The paper "A two-staged Approach for Localization and Classification of Coral Reef Structures and Compositions" [4] which is submitted in 2019 had the best performance in the first edition of coral reef. Their proposed model achieved top results in the competition which was based on YOLO a state-of-the-art deep learning approach. They used a two staged

method with two classifiers, the first classifier classifies the tiles of the grid and the other classifies the found boxes. For the training dataset they had 246 images with 6670 annotated substrate which is classified into 13 classes. Since this was in the first edition the number of images is comparatively less. In the pre-processing phase they used the image sharpening and data augmentation. Since the dataset is small, the augmentation helped to increase the number of training images and the sharpening method is applied to increase the over image quality and increasing the pixel definition.

The Model they used for the segmentation is YOLO, a deep learning algorithm which is used for real time object detection using convolutional neural networks. This algorithm is popularly used in object detection tasks due to its speed and accuracy. The task they did in two stages. In the first stage the image is divided into different grids. In their study they rescaled the images into 608x608 and divided the girds in the dimension of 12x12, so that it can located even the smallest boxes without background. In the next stage, the features of each tile is extracted for all the training images and these features were used to for the training the binary classifier which predicts whether the block is a part of coral area or not. For the binary classification the K-Nearest-Neighbour with K value as 15 is used and obtained a black and white image as prediction. After that the same features that they extracted before is used to create the bounding boxes using CNN. The CNN model they used is comparatively small with one convolutional layer, maxpooling layer and RELU as activation function. In order to avoid overfitting, the dropout method is also implemented into the CNN and finally since it is a multiclass classification a SoftMax function is used in the output layer of the model. The models were trained with a batch size of 100 and up to 1000 epochs. The evaluation method they used is Mean Average Precision and IoU. Even though the model they developed had the top performance in that edition, the results were moderate due to the class imbalance and overlapped bounding boxes. The CNN and K-NN models had almost the same performance and YOLO with statistical methods had the best results with an average M.A.P value of 0.243 with an IoU >= 0.5. The image sharpening didn't improve their model performance due to the high noise.

Takeaways and questions arised from the study:

- How to over come the overlapping of bounding boxes?
- How increase the quality of the input image without creating noises?

- How to deal with the class imbalance?
- Will increase in the size of training data will improve model performance?
- The predictions using YOLO architecture outperforms the conventional machine learning methods
- Since YOLO is based on regression, is there any other models that can do the classification better than YOLO?

In the same year another team, which includes the researchers from Filament AI and University of Essex came up with another approach by using a more sophisticated deep learning architecture called DeeplabV3 [5]. The method they followed is entirely different form the previous paper. Instead of using bounding boxes in order to classify the substrates, they used semantic segmentation method which is based on pixel wise parsing were all the pixels in the image is classified into individual classes. The dataset they used is same as the previous team and split the data into 85:15 for training and validation. The data was split not randomly but on a per image basis, this will helped them to compensate with the inter-image classification problem and led to better results in the evaluation stage. The image pre-processing task includes the data argumentation with flipping and cropping, with a cropping size of 400x400 and 1400x1400. After that for standardising the sizes, the images were then rescaled into 256x256. The flipping is done in both vertical and horizontal directions. The model they used is DeepLabV3 a Convolutional Neural Network architecture and pretraining the model on ImageNet dataset. The authors used ResNet-50 and ResNet-101 as network backbone for extracting the features and the features extracted is then combined by feature pyramids using the atrous convolutions. Then the individual output is concatenated into a new feature map. In order to address the problem of class imbalance they weighted loss function approach were the loss functions are weighed based on the class distribution. The loss function used is cross-entropy and the weights are calculated using the formula:

$w(C) = 1/log(\alpha + p(c)$

The model is trained for 50 epochs and a batch size of 32, with two images in each batch and 16 crops on each image. For the prediction phase they implemented a sliding window approach, in which the images used for prediction is divided into 112 overlapping sections and scaled them into 256x256, then fed into the model for prediction. The predicted output is

rescaled them into their original sizes and align them in position to create a 4032x3024 image. There were many post processing methods used in their paper like morphological analysis, flood fill, Douglas-Peucker algorithm for polygon approximation etc. The evaluation method they used is mean IoU with the value of the same is calculated for individual classes. The maximum IoU values they got is 21.9% for hard coral mushroom followed by hard coral boulder with a value of 16.59%.

Takeaways and questions arised from the study:

- Training the model in different batch size and epochs will improve the results?
- Using an image filter, thereby increasing the quality of the image will help the model performance?
- Different backbones for extracting feature maps will have any effect?
- Research more on cropping of the images and bootstrapping
- Any other methods to deal with the imbalance of the class distribution, since the method they used didn't have much effect on prediction

We are also discussing one more paper in this section which is based on a different model compared to the above two. The paper " Automatic Coral Reef Annotation, Localization and Pixel-wise Parsing Using Mask R-CNN" [6] by Lukas Soukup is submitted in the 3rd edition of the ImageCLEF coral which proposed a method based on Mask R-CNN object detection and segmentation framework. In the 3rd edition more images were added to the training data which consist of 1052 train images and 485 test images. In his paper the approach he followed is to create a bounding box around the substrates which is specified by 4 numbers and in the second task he implemented the binary segmentation of the substrates. During the data pre-processing task, the data augmentation is applied on the dataset with a random horizontal flip, random brightness, and contrast variations. The model used for the classification is Mask R-CNN which is developed based on faster R-CNN by creating a branch that will predict the output mask thereby having a better pixel to pixel alignment. The model operates in two stages, the first stage is a Region Proposal Network (RPN) which identifies the different objects present in the given image and in the second stage the model predicts the output as a binary mask for each region of interests along with predicting the

classes. The main advantage of using Mask R-CNN is that it is easy to implement and can be used over a variety of architectural designs. The evaluation criteria used in his study is mean average precision with an IoU value greater than or equal to 0.5. The maximum value of the M.A.P got in his study is 0.35 for instance segmentation.

All the papers mentioned above followed different approaches and the results were comparable. It is noticeable that the results got by all these models and methods were so low compared to other segmentation tasks. This is due to the quality of the images, the region in which the images were taken, different light conditions, imbalance in the classes etc. The pre-defined CNN architecture gave the better results in all these papers and for the evaluation the mean average precision along with IoU is mostly used. Due to the presence of large number of substrate in a single image (up to 45 to 50) it will be difficult to localise them using bounding boxes, since it will overlap over each other and makes it hard to differentiate. So our approach is to do the multi class semantic segmentation using the advanced version of deeplab which is deeplabV3plus as our model. In the image pre-processing phase, there must be an enhancement method that will improve the overall image quality. It is also noticeable that the scaling of the image didn't affect the model performance but reduces the run time and model complexity. The approach of our studies is to address the issues that the participants faced during the previous editions and an attempt to create a better method which will help in the future studies. We are trying to find answers for the following questions in this paper:

- RQ1:How increase the quality of the input image without creating noises?
- RQ2:How to deal with the class imbalance?
- RQ3:Will increase in the size of training data will improve model performance?
- RQ4:How the data augmentation affecting the model performance?
- RQ5:Training the model in different batch size and epochs will improve the results?
- RQ6:Is it possible to came up with a single model for predicting the images from different locations?

CHAPTER 3

Methodology

The project is divided into two parts: The first one is the annotation task, creating the image masks for training the model and the second is the segmentation task were we feed the train images and annotated images into our CNN model. The model is then evaluated, and sample predictions are made. By analysing the previous studies, we identified that semantic segmentation is the best approach compared to the localisation and bounding box method. We can approach it in two ways: in the first method we could select one class each time and mark it as ROI and in the second approach we could select the whole 13 class in a single go and make the predictions for all the classes in a single image. The first one is a binary class classification task and the second one is multiclass classification. Comparing the two the binary classification is much easier than multiclass segmentation, in our project we are trying the multiclass semantic segmentation in which each substrate class is denoted by different colour codes and the target colour codes are labelled down for the classification task. Due to the system limitations a subset of images were generated from all the 5 locations with a sample size of 50. The individual masks are created for all the samples in separate folders and model is trained and evaluated against each. The results were compared, and an analysis is made based on the outcome and finally a new sample set is created from these 5 samples with 10 images from each folder. The model is trained on this new sample data and predictions are made also, using the same model prediction made against the previous sample sets and results were compared. This is done in order to identify whether it's possible to came up with a single model that can do the predictions on the entire dataset, because the images from

the different location varies in quality, colour, no of classes etc. For evaluating the results we are using the mean IoU values not from the values of individual images but for the overall values of the validation images.

All the codes for both the annotation and segmentation is written in python, The opencv library is extensively used for the annotation task and also libraries like scikit learn, numpy, pandas, tensorflow, keras etc were used through out the code. The details structure of the project is explained in the upcoming sections.

3.1 Dataset

The dataset consist of 1363 images which collected by using an underwater multicamera system developed by University of Essex. A movable 5-camera array was used to collect the photographs for each model as it traversed the landscape. The images frequently have a 60% overlap and are likely to contain similar landscape elements taken from different perspectives. After aligning the images with Agisoft Metashape, they were processed with "medium" processing parameters to produce a 3D textured model. The co-ordinate data which is given in csv format is created by the marine biology post graduate students at the university and again these annotations were cross verified by a marine biologist who is an expert in coral reef structures. There are two csv files: one has the co ordinate values that can be used to draw boundary boxes around substrate classes, which is mostly used for localisation and annotation task. The second csv file consist of the co-ordinate values to draw polygons around the individual substrates, which will define the corals more accurately in the image. We are using these coordinate values for creating the mask images and later for pixel-wise parsing.

And in the case of image data, the images are arranged in 6 different folders based on the location, the locations in which the data taken are:

- K1, Kaledupa, Indonesia (K1-20180712-01)
- PK, Hoga Indonesia (PK-20180714-01 PK-20180729-02)
- Keke, Spermonde Archipelago, Indonesia (20180406-spermonde-keke)
- Curieuse Island, Seychelles (20190417-seychelles-BL)

- Cabrits, Dominica, Caribbean (20170803-dominica-cabrits)

The same type of substrates can be seen in different size, colour, shape etc , it makes the task more difficult for classification. Some images includes the white measuring tape overlapping with the substrate object and can be removed while creating the image masks. The quality of the images varies over the entire dataset, some are so blurry, some have very low colour balance etc. For the last edition in 2022, all these images were reviewed again in order to correct the errors.

The 13 classes are:

Hard Coral Branching	Soft Coral
Hard Coral Submassive	Soft Coral Gorgonian
Hard Coral Boulder	Sponge
Hard Coral Encrusting	Sponge Barrel
Hard Coral Table	Fire Coral Millepora
Hard Coral Foliose	Algae Macro
Hard Coral Mushroom	

Table 3.1: Coral Subclasses

Sample images from dataset:

Figure 3.1: Train Image Sample: A sample image from Dominica-Cabrits

The sample images shown here are from three different locations, it is noticeable that the images vary in texture, colour etc and also the image quality is not same throughout the dataset.

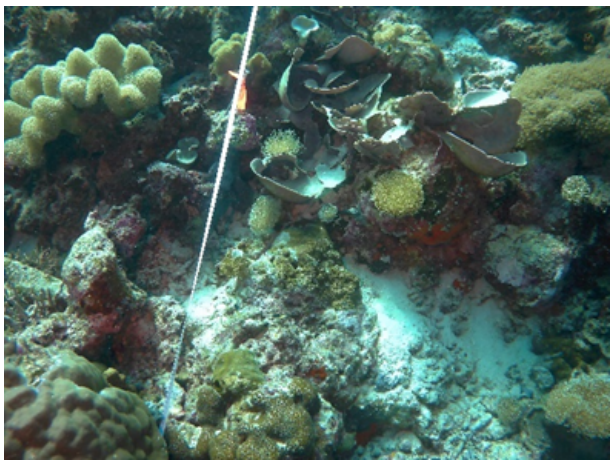

Figure 3.2: Train Image Sample: A sample image from PK-20180714-01

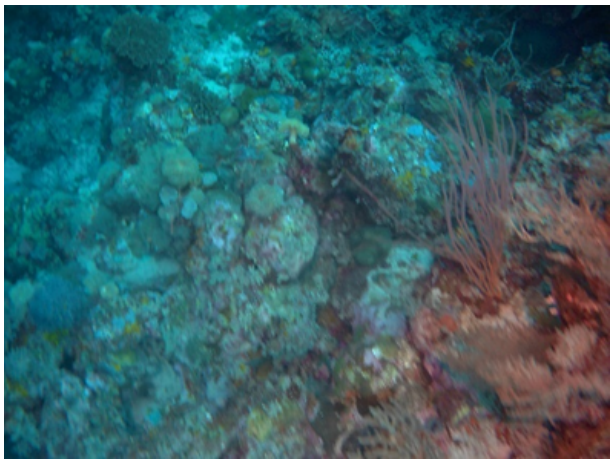

Figure 3.3: Train Image Sample : A sample image from K1-20180712-01

3.2 Annotation

For the annotation we are using the polygon coordinates instead of drawing the bounding boxs, so the csv file which contains the coordinates to draw the polygons is used for the annotation task. First of all, we will combine all the train images together from the different locations into a single folder and the final folder contains 1363 number of images. Then we have to convert the csv file into our required format and again convert it into a data frame using pandas library. The glob and OpenCV libraries are used to read and write the images in the folder and the corresponding coordinate values of each image is fetched using a for loop.

In the previous papers they draw the mask over the top of the image itself but in this study, we are following a different method. We create a blank image with black background with the same shape as the train image. Then coordinates corresponding to the each image is drawn in the blank image that we created and the final masked image is saved in a separate folder with the same name as of the train image in "tif" extension. Since we have 13 classes we are using 13 colour codes for each class, that is each colour in the mask other than black represents a substrate class which is our ROI.

3.2.1 Data Pre- Processing

The first task for creating the masked images is to convert the csv file into our required format. The sample of the raw data is shown in the below image:

COLUMN1	COLUMN2	COLUMN3	COLUMN4	COLUMN5	COLUMN6	COLUMN7
2018_0714_112502_024	0	c_sponge_barrel	1	2037	2648	2114
2018_0714_112502_024	1	c_soft_coral_gorgonian	1	2639	2283	2668
2018_0714_112502_024	2	c_soft_coral	1	3834	3008	3834
2018_0714_112502_024	3	c_soft_coral	1	49	1836	150
2018_0714_112502_024	4	c_soft_coral	1	994	1838	803
2018_0714_112502_024	5	c_hard_coral_submassive	1	983	550	933
2018_0714_112502_024	6	c_hard_coral_mushroom	1	1175	718	1254
2018_0714_112502_024	7	c_hard_coral_submassive	1	1218	838	1259
2018_0714_112502_024	8	c_hard_coral_boulder	1	1113	455	1107
2018_0714_112502_024	9	c_soft_coral	1	301	843	322
2018_0714_112502_024	10	c_soft_coral	1	10	1274	110

Figure 3.4: Raw Dataset

We can divide the dataset into 4 parts [ref Fig 3.4]. The first part represents the image

name which is given in column no 1 and the column 2 represents the no of substrates present in the image, it should keep in mind that the number starts from zero instead of one. Column 3 gives the name of the class in which each substrate belongs, and the coordinates to draw polygons around the substrates is given from column 5. Since we are drawing the polygons in a 2D the coordinate values should read in pairs, for example column 5 and column 6 should be read as pairs in which the first column represents the y co-ordinate value and second one represents the x co-ordinate value of a particular point. We are ignoring the fourth column since it doesnât have any use for our annotation and the final dataset contains total 918 columns and 31,518 rows in total.

Since we are using the OpenCV to draw the masks, co-ordinate values should be given as an array of lists. In order to do that first we create two empty lists then using a for loop we iterate over the columns and the values in the first and second columns are stored in two separate variables. The variables is then appended into the first empty list as pairs and finally another for loop is used to append these paired list in to the second empty list that we created.

Sample output after first iteration : 234,355

Sample output after second iteration: [[234,355]]

The iteration continues till all the values in the columns are paired and added in to the final list. The list is then converted into pandas data frame and finally stored as an excel file. The output is shown below:

Co-ordinate Points
[[3834,3008],[3834,3008],[3712,2941],[3668,2906],[3668,2906]]
[[3221,2530],[3146,2610],[3067,2687],[3016,3001],[3016,3001]]
[[3708,2689],[3602,2625],[3568,2548],[3491,2539],[3314,2508]]

Table 3.2: Co-ordinate Points : Sample data

The points are now arranged in as a list of lists, so we need to unpack that first. For that we use normal excel functions to remove the first and last element of the cell. The output is shown in table 3.3.

Finally, we have the co-ordinate values in our required format, now we need to add this as a new column to our original dataset. The final dataset shown in Figure 3.5 has five columns with column names a1, a2, a3, a4 and a5. The column a1 contains the image name and the

Co-ordinate Points
[3834,3008],[3834,3008],[3712,2941],[3668,2906],[3668,2906]
[3221,2530],[3146,2610],[3067,2687],[3016,3001],[3016,3001]
[3708,2689],[3602,2625],[3568,2548],[3491,2539],[3314,2508]

Table 3.3: Co-ordinate Points : Sample data

a1	a2	a3	a4	a5
2018_0714_112502_024	0	c_sponge_barrel	1	[2037, 2648],[2114, 2567],[2231, 2434],[2231.0, 2434.0],[2249.0, 2372.0],[2249.0, 23
2018_0714_112502_024	1	c_soft_coral_gorgonian	1	[2639, 2283],[2668, 2404],[2631, 2517],[2639.0, 2643.0],[2637.0, 2738.0],[2511.0, 28
2018_0714_112502_024	2	c_soft_coral	1	[3834, 3008],[3834, 3008],[3712, 2941],[3668.0, 2906.0],[3668.0, 2906.0],[3719.0, 27
2018_0714_112502_024	3	c_soft_coral	1	[49, 1836],[150, 1766],[297, 1783],[267.0, 1680.0],[285.0, 1571.0],[450.0, 1536.0],[4
2018_0714_112502_024	4	c_soft_coral	1	[994, 1838],[803, 1685],[614, 1534],[614.0, 1534.0],[455.0, 1543.0],[322.0, 1434.0],[
2018_0714_112502_024	5	c_hard_coral_submassive	1	[983, 550],[933, 622],[917, 693],[917.0, 693.0],[875.0, 757.0],[967.0, 794.0],[1096.0,
2018_0714_112502_024	6	c_hard_coral_mushroom	1	[1175, 718],[1254, 718],[1254, 718],[1321.0, 761.0],[1321.0, 761.0],[1355.0, 816.0],[1
2018_0714_112502_024	7	c_hard_coral_submassive	1	[1218, 838],[1259, 821],[1259, 821],[1358.0, 844.0],[1358.0, 844.0],[1351.0, 895.0],[1
2018_0714_112502_024	8	c_hard_coral_boulder	1	[1113, 455],[1107, 391],[1107, 391],[1134.0, 344.0],[1189.0, 409.0],[1189.0, 409.0],[1
2018_0714_112502_024	9	c_soft_coral	1	[301, 843],[322, 793],[324, 791],[402.0, 788.0],[460.0, 804.0],[460.0, 804.0],[540.0, 7
2018_0714_112502_024	10	c_soft_coral	1	[10, 1274],[110, 1207],[194, 1313],[194.0, 1313.0],[165.0, 1403.0],[109.0, 1500.0],[11
2018_0714_112502_024	11	c_soft_coral	1	[1966, 1755],[2108, 1760],[2201, 1793],[2243.0, 1830.0],[2236.0, 1970.0],[2238.0, 19
2018_0714_112502_024	12	c_soft_coral	1	[2440, 1537],[2507, 1502],[2659, 1521],[2659.0, 1521.0],[2728.0, 1601.0],[2712.0, 16

Figure 3.5: Final Dataset

number of substrates in that image is given in a2. Class category of the each substrate is given in column a3 and finally the coordinates to draw the polygons around the substrate is in column a5.

3.2.2 Creating the Masks

At this stage we have the dataset in the required format, now we will create a function to draw the mask for the images. For that first we will convert the excel file into a data frame using pandas as shown in figure 3.6.

	a1	a2	a3	a4	a5
0	2018_0714_112502_024	0	c_sponge_barrel	1	[2037, 2648],[2114, 2567],[2231, 2434],[2231.0...
1	2018_0714_112502_024	1	c_soft_coral_gorgonian	1	[2639, 2283],[2668, 2404],[2631, 2517],[2639.0...
2	2018_0714_112502_024	2	c_soft_coral	1	[3834, 3008],[3834, 3008],[3712, 2941],[3668.0...
3	2018_0714_112502_024	3	c_soft_coral	1	[49, 1836],[150, 1766],[297, 1783],[267.0, 168...
4	2018_0714_112502_024	4	c_soft_coral	1	[994, 1838],[803, 1685],[614, 1534],[614.0, 15...

Figure 3.6: Dataframe

We also checked the number of substrates against each classes:

Class	No of Substrate	% Value
soft coral	7745	25%
hard coral boulder	7355	23%
sponge	6077	19%
hardcoral branching	3112	10%
hard coral submassive	2618	8%
algae macro or leaves	1867	6%
hard coral table	920	3%
sponge barrel	606	2%
hard coral encrusting	380	1%
hard coral mushroom	335	1%
hard coral foliose	233	1%
hard coral gorgonian	169	1%

Table 3.4: Persentage of substrate in each class

By analysing table 3.4, soft coral has the most no of substrates followed by hard coral boulder and sponge. The class imbalance is also clear from the data. The first 6 classes together consist the 91% of the total substrates and the last 5 classes has only 5% of the overall data.

For creating the mask first, we need to assign specific hue values against each classes, a dictionary is created with class names as keys and hues as values. Then using a for loop we fetch the classes and coordinate for each trainimage and created a data frame with that data. Now a blank image is created against each train image with the same name and shape. Then using the CV2.FILLPOLY function in OpenCV we draw the masks in the blank image and saved it with tif extension on a separate folder in the drive using CV2.WRITE. Since OpenCV read images as BGR values we converted the masked images into RGB images using CV2.COLOR_HSV2RGB function before writing it in the folder. We using the tif extension here because it has predominantly lossless compression that helps to retain the quality of the image and retains its colour and depth compared to the png and jpeg format.

Sample mask images:

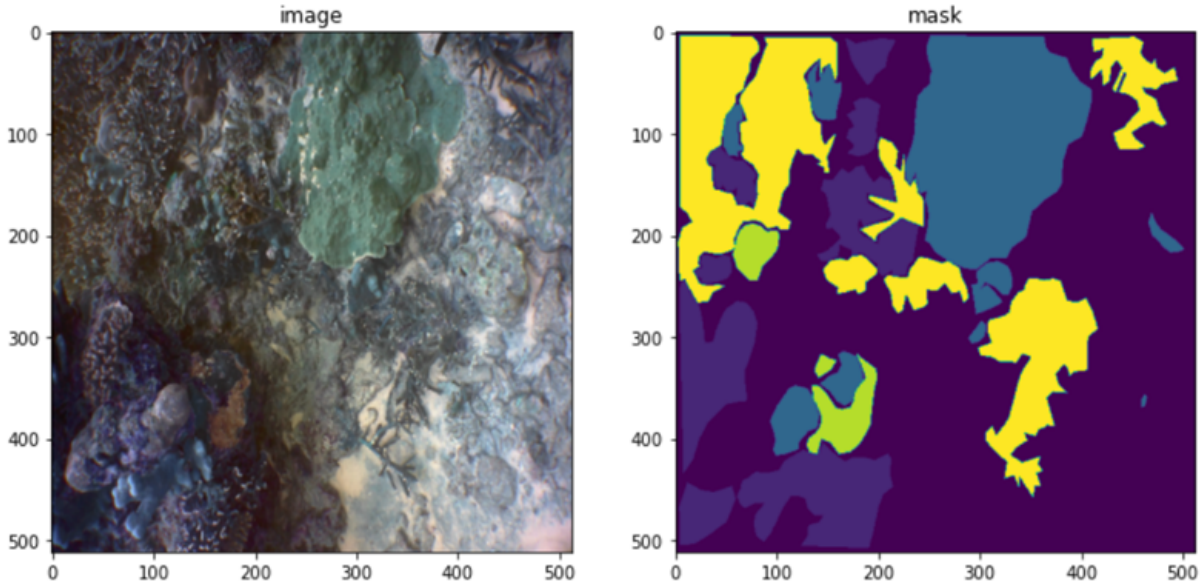

Figure 3.7: Sample mask image : Image from 20190417-Seychelles-BL

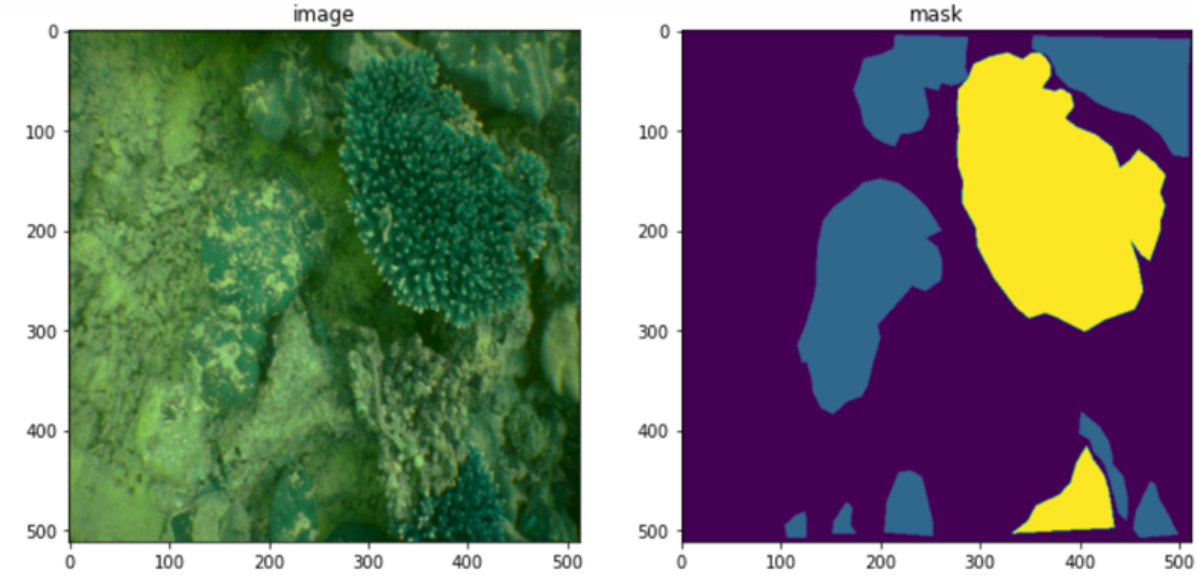

Figure 3.8: Sample mask image : Image from 20190417-Seychelles-BL

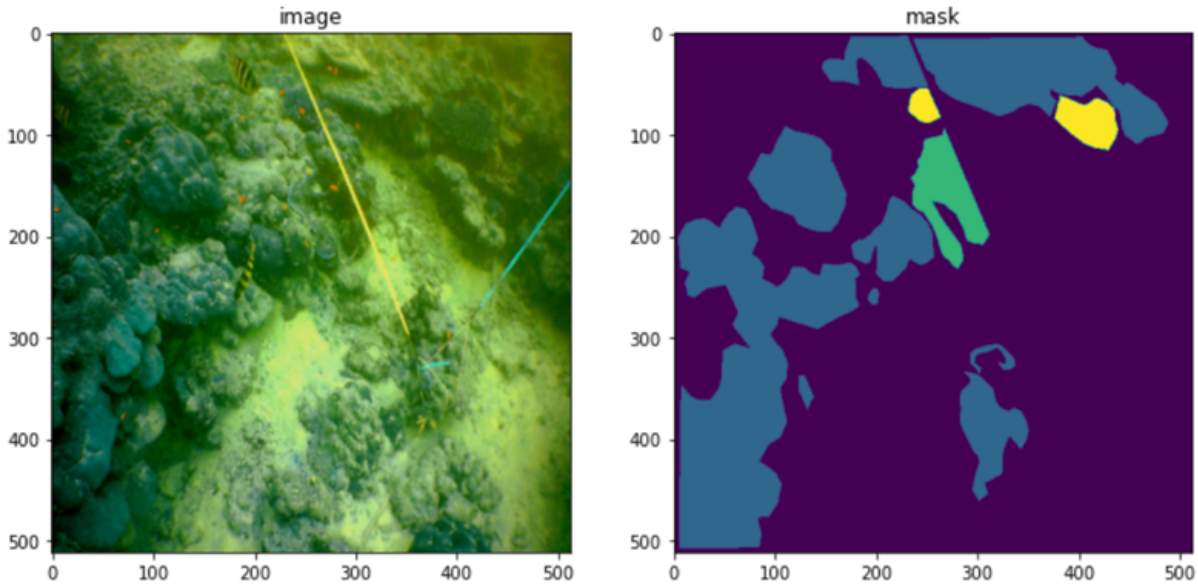

Figure 3.9: Sample mask image : Image from PK-20180714-01

3.3 Semantic segmentation

In segmentation task each pixel in an image is mapped into different objects. A segmentation mask is created by grouping the pixels in a localised image. There are different types of segmentation tasks such as instance segmentation, semantic segmentation and panoptic segmentation. In semantic segmentation all the images that belongs to a particular class is represented in a single mask and for the instance segmentation each object in the class will again segmented into different masks. Panoptic segmentation is the combination of above two which segments the image both by class and instance. The application of the semantic segmentation is ranging from autonomous driving, medical imaging, background removal etc. Here in this project, we are following the semantic segmentation method for the pixel-wise parsing task[7].

The segmentation task can be done using deep learning methods like Convolutional Neural Networks (CNN) and is widely used in computer vision tasks. There are many CNN methods developed to perform the segmentation tasks, some of the popular models are :

- Fully Convolutional Neural Network (F CNN)
- UNet: A modification of F CNN, mainly used for medical imaging
- Pyramid Scene Parsing Network (PSPNet) : Designed for scene parsing

- DeepLab: Developed by google which applies atrous convolution for up sampling

  In this project we chose the updated version of DeepLab called DeepLabV3+ as the core architecture since this model performed well in the previous editions.

3.3.1 Model Architecure:DeepLabV3+

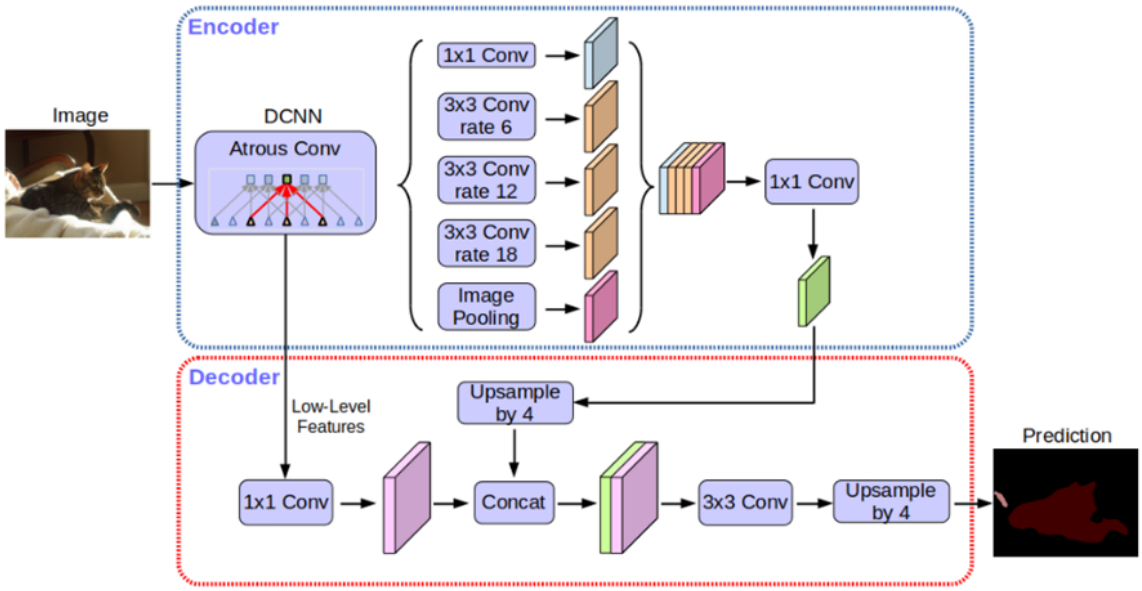

Figure 3.10: DeeplabV3 : Model Architecture [8]

The deeplab is specifically designed for semantic segmentation task, the model is designed and developed by the researchers from google back in 2016. The latest version of deeplab is deeplabv3+ which consist of an encoding and decoding phase in which it has an improved Atrous Spatial Pyramid Pooling (ASPP) module that uses batch normalisation and pixel-level features. The encoding phase uses a network backbone to extract features from the images and then by using the ASPP network the feature map size is controlled in order to classify the pixels corresponding to their classes. The output of the ASPP is then passed to a 1x1 convolutional layer to retain the actual size of the image which is the segmented output mask [8].

The architecture can be divided into 3 stages:

- Encoding phase

- ASPP
- Decoding phase

The architecture has a deep encoder in the input phase which is used to extract the feature maps from the input image. The dimension of this feature maps is converging over the layers and will help the model to capture a longer range of features from the image. A backbone network like ResNet50 or ResNet101 is used with pre trained weights for extracting the features in the encoder. The extracted features is then given as input to an ASPP layer. This layer will classify each pixels in the image based on their class category.

The ASPP layer uses astrous convolution or dilated convolutions to retrieve muti scale-context information's from the encoded image. This is an effective method to enhance filed of view of the filters without increasing the computational complexity. For that, it uses a parameter called dilation rate or astrous rate which is similar to the conventional convolutions except zeros are inserted between 2 consecutive filters along each spatial dimension. It consist of 4 convolutional layers and 1 image pooling layer. The main limitation of the convolutional layers are they will always record the exact position of the feature maps in the input image, so a slight variation in the input image which may happen during data augmentation may results in different feature. In order to avoid that down sampling method is used were a low resolution of the input image is created with out loosing any feature elements. The down sampling can be done by using convolutional layers by altering the stride values across the image or by using a pooling layer. Here we are using an Image pooling layer on the top of the 4 convolutional layers.The image is up sampled again after pooling and convolutional layers are applied on it. It passed through one 1x1 convolutional layer followed by three 3x3 convolutional layers. The dilation rate of these 3 layers are 6,12 and 18 respectively and uses relu as activation function.

The image is then enter into the decoder phase where the object feature and details are recovered back again. This is done by using a 1x1 convolutional layer followed a concatenation and two 3x3 convolutions. An up sampling is done in the output layer with same padding and softmax activation function.

3.3.2 Squeeze-and-Excitation Block (SENets)

A SENet block is added in to the decoder phase of the architecture for channel independencies without any computational cost. The idea behind SENets is to allow the network to adjust the feature map weights by adding parameters to each convolutional blocks. In normal case, model weighs the channels equally while creating the output maps.

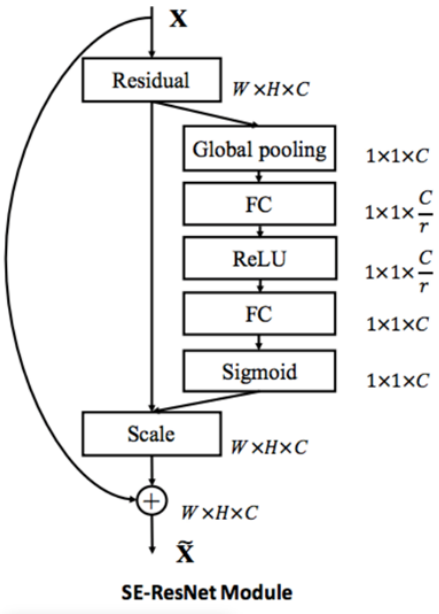

Figure 3.11: SE ResNet Module [9]

The input to the SENet is a convolutional block and number of channels and then by using the global pooling method we squeeze all the channels to a single numeric value. Then it has a fully connected layer with relu activation in order to add the nonlinearity and to reduce the output channel complexity. The second fully connected layer with sigmoid function will create a smooth gating function for each channel. Both the FC layers uses 'he normal' as kernel initialiser and finally based on the result it weighs each feature maps of the convolutional block [9].

3.3.3 Image Filter

When the dataset is loaded, the images were passed through an image filter for smoothening and removing the noises. There are different build in filters available in OpenCV here we are using the bilateral filter. This filter is very effective in removing the noise without affecting

the sharpness of the image. It's a kind of gaussian filter which replaces the intensity values of each pixel compared to the weighted values of the neighbouring pixels. In standard gaussian filter only the nearby pixels are considered, it will not consider their intensities and will not check whether it is an edge pixel or not. This results in blurring the edges there by reducing the detailing of the image. One more advantage of the bilateral filter is that, it will not consider the nearby pixels based on the location but also the photometric range of those pixels.

3.3.4 Label Encoding

Since we have the class labels in pixel values we need to convert the class labels into numerical values. There are total 13 classes in the image including the background so there will be 13 pixel values in the mask image. For converting that into machine readable form we are manually assigning numerical values to each pixel ranging from 0 to 12. The values corresponding to each pixels are shown in the table [5]. These values are again converted into binary class matrix using (to categorical) function.

Pixel Values	Class Labels
[0,0,0]	7745
[0,0,255]	1
[0,170,255]	2
[0,255,0]	3
[0,255,84]	4
[170,255,0]	5
[255,0,0]	6
[0,85,255]	7
[255,84,0]	8
[255,169,0]	9
[255,254,0]	10
[0,255,169]	11
[255,0,85]	12

Table 3.5: Class Labels

3.3.5 Compiling the Model

After the label encoding the dataset is split into train and test data in 80:20 ratio. Since we are using a sample of 50 images, we have 40 images for training and 10 images for testing. For compiling the model the optimiser used is "adam" and the loss we used here is "categorical crossentropy". Adam or adaptive Moment estimation is an algorithm used for gradient decent and it combines both the "gradient descent with momentum" and the "RMSP" optimizers. The mathematical intuition of the adam optimizer is shown below [10].

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)\left[\frac{\delta L}{\delta w_t}\right] v_t = \beta_2 v_{t-1} + (1-\beta_2)\left[\frac{\delta L}{\delta w_t}\right]^2$$

Figure 3.12: Adam Optimizer: Mathematical Expression

The rate of gradient decent is controlled by minimising the oscillations while converging into the global minima by bypassing the local minima points along the way. There by reaching the global minima point more efficiently.

Since we are having a multiclass classification problem we are using categorical cross entropy as loss function, it will train the model to output a probability over the number of classes in each image. The model is then trained on a batch size of one and for 20 epochs. The batch size is kept as one in order to reduce the run time complexity due to system limitations.

3.3.6 Data Augmentation

Data augmentation is the process of creating a new training dataset from the existing one by applying techniques like flipping, shifting, rotating, rescaling etc. This is mostly applied to a dataset when the no of training images are less and it also helps to avoid overfitting thereby improving the model performance. Here the data augmentation techniques we used are:

- Rotation: Simply rotating the image in clockwise direction by a particular angle, the rotation angle applied here is 20 degrees.
- Shifting: The entire pixel of an image is shifted from one position to another, we can do the shifting in both horizontally and vertically. We applied both in our dataset.
- Flipping: In this operation the actual image is flipped horizontally or vertically over one axis of the image. Here we used horizontal flip

- Scaling: Scaling is resizing the image, a zoom range of 0.3 is applied over all the images

The fill mode used here is constant, which covers the blank areas in the augmented image with black background. There are different libraries for doing the augmentation task, some copies the images and write it on a separate folder prior to the training stage. Other libraries randomly set the transformation during the training phase of the model, due to this the space in which the optimiser searching is increased. So, it does not require any extra space for storing the augmented dataset. We are using the second method here.

3.3.7 Model Evaluation

After validating the model on the test datatset the performance of the model is evaluated. The segmentation models are normally evaluated using a mertic called Intersection Over Union (IoU), also called Jaccard index. In this method the percentage of overlap between the target mask and prediction mask are calculated. In other words it measures the number of pixels common in between the prediction mask and target mask divided by the total no of pixels present in total masks. The mathematical representation of IoU is shown below [11].

$$IoU = \frac{(target \cap Prediction)}{(target \cup Prediction)}$$

Where the intersection denotes the number of pixels present in both testing image and predicted image and the union represents the total no of pixels found in both images. For multiclass semantic segmentation the IoU score for each class is calculated separately and a mean value of all these scores are given as output. The following equation is used to calculate the IoU values of the individual classes.

IoU = true positives / (true positives + false positives + false negatives)

Here in order to calculate the Mean IoU we are using the keras build in mean IoU function. It will take the number of classes as argument and this value is used to create a confusion metrix of size [number of classes , number of classes]. The metrix is weighted by the sample weight and the final IoU value is calculated from it. If we didn't mention the sample weight the default value is 1. We are using the default value here and normally IoU value above 0.5 is considered as good prediction .

Figure 3.13: IoU scoring Example [11]

Along with the mean IoU the pixel accuracy is also used as an evaluation matrix during the training phase and this value is normally calculated globally along all classes. The equation for calculating the accuracy is shown below:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}.$$

This metric is not an accurate method to evaluate the performance of the semantic segmentation models, since sometimes it will give misleading results when the area of masks presented in an image is very small compared to the background.

CHAPTER 4

Results

Overall, the results we got in this study is moderate compared to the previous editions. Although we have tried different methods and parameters to identify how it will affect the model performance. The result against each method is tabulated and also the predictions are visually represented for better understanding and we are trying to answer the research questions we had during our literature survey by analysing the output we got.

- RQ1: Does the model performance change with the number of epochs?

In order to identify that we trained our model at different epochs with a batch size of 1 due to run time complexity. The results are visualized below:

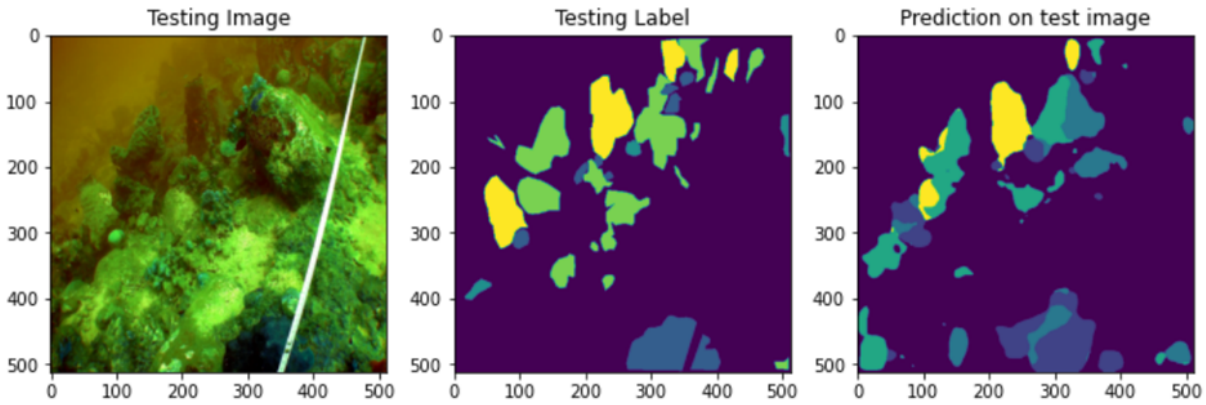

Figure 4.1: Output at 5 epochs

- RQ2: Does increasing the size of the dataset by augmentation is helpful?

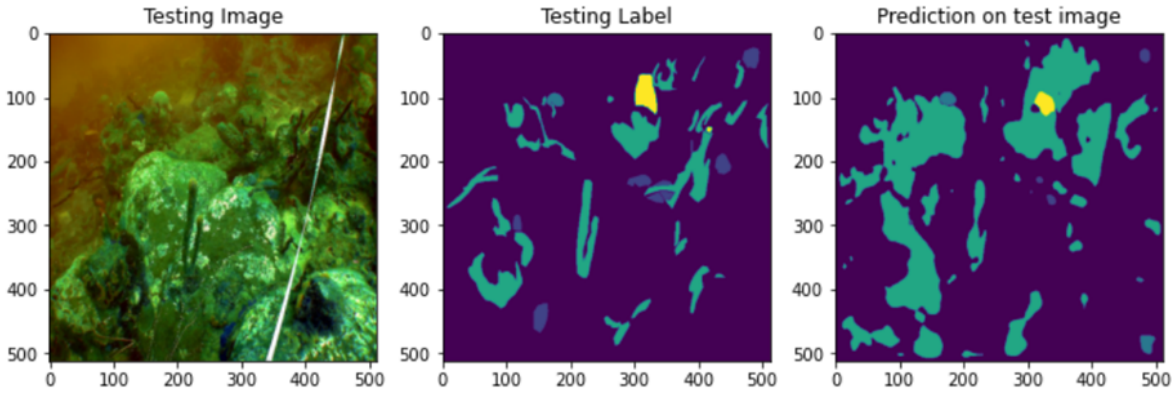

Figure 4.2: Output at 10 epochs

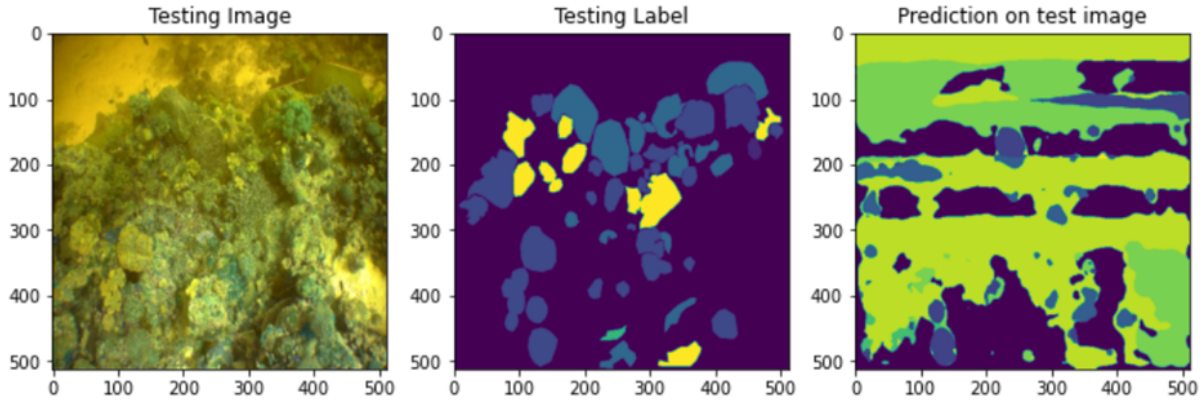

Figure 4.3: Output at 20 epochs

As discussed in the previous section we used different augmentation techniques in our train image data. The output with and without augmentation is given here:

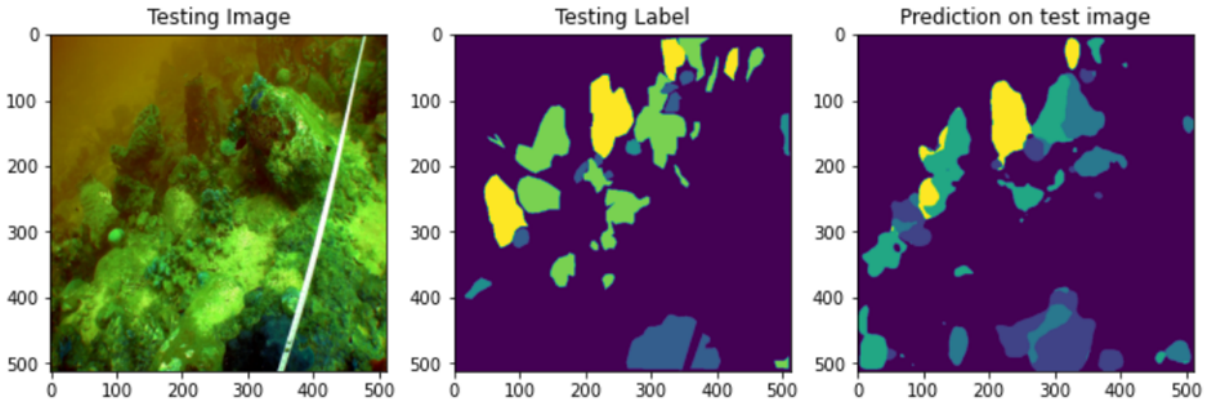

Figure 4.4: Prediction Without Augmentation

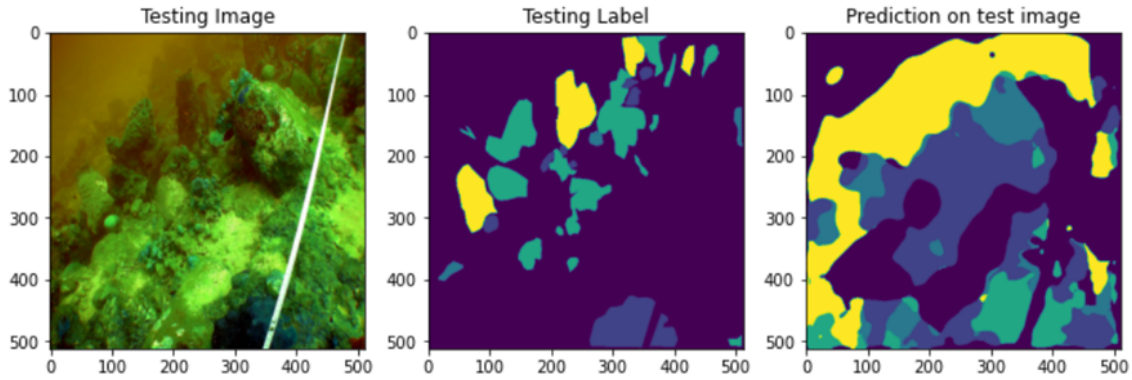

Figure 4.5: Prediction With Augmentation

	Mean IoU
Without augmentation	0.152
With augmentation	0.141

Table 4.1: Augmentation Results

- RQ3: What is the effect of different loss functions in model performance?

Here we are comparing the two loss functions: Categorical cross entropy and dice loss.

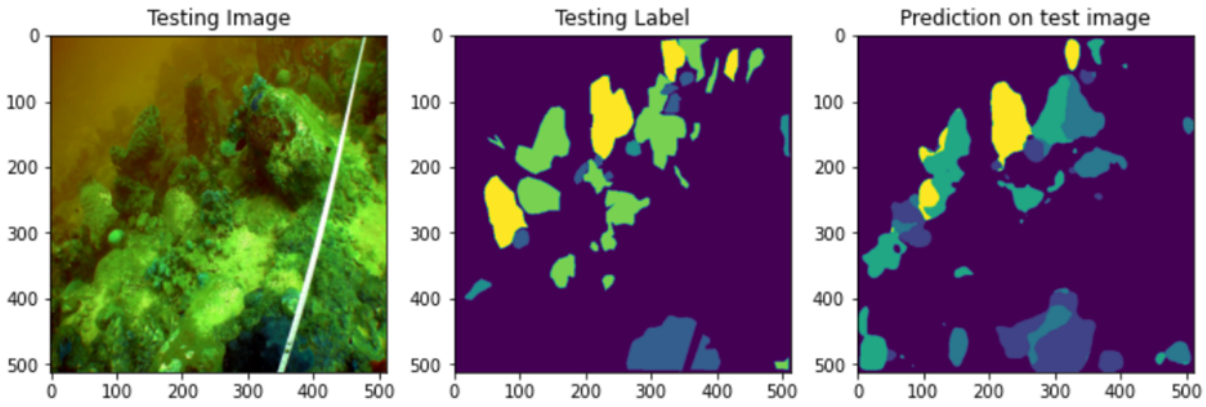

Figure 4.6: Prediction using Categorical Cross Entropy

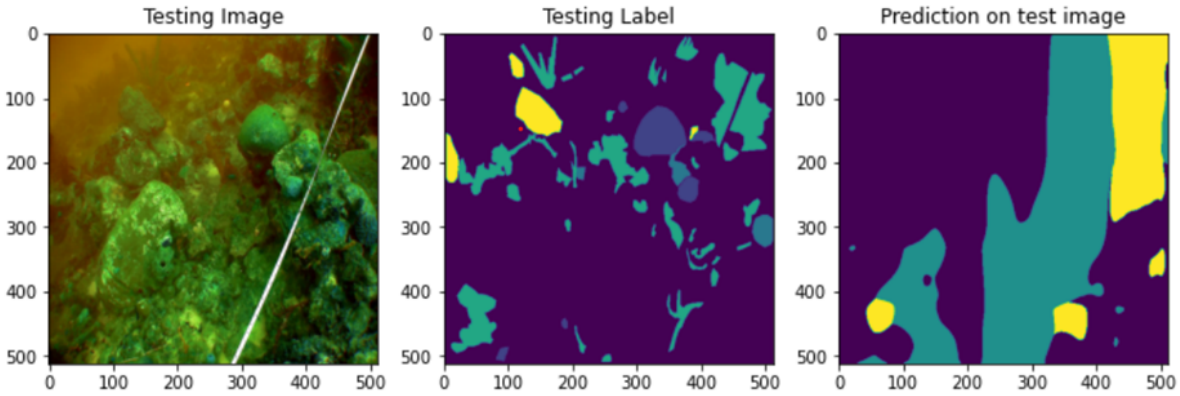

Figure 4.7: Prediction using Dice Loss: With Class Weights

	Mean IoU
Categorical Cross Entropy	0.152
Dice loss: Default weight	0.0059
Dice loss: Class weight	0.113

Table 4.2: Results With Different Loss Functions

- RQ4: How the model performance changes with the input image quality?

Since the images are taken in underwater the image quality varies from one location to another. In the previous sections we discussed this in detail. So in order to remove the noise and smoothing the image we passed the input image through a bilateral image filter. The results are displayed in fig [4.8 and 4.9] with the corresponding mean IoU values [table : 4.3].

- RQ5: Is it possible to came up with a single model for predicting the images from different locations?

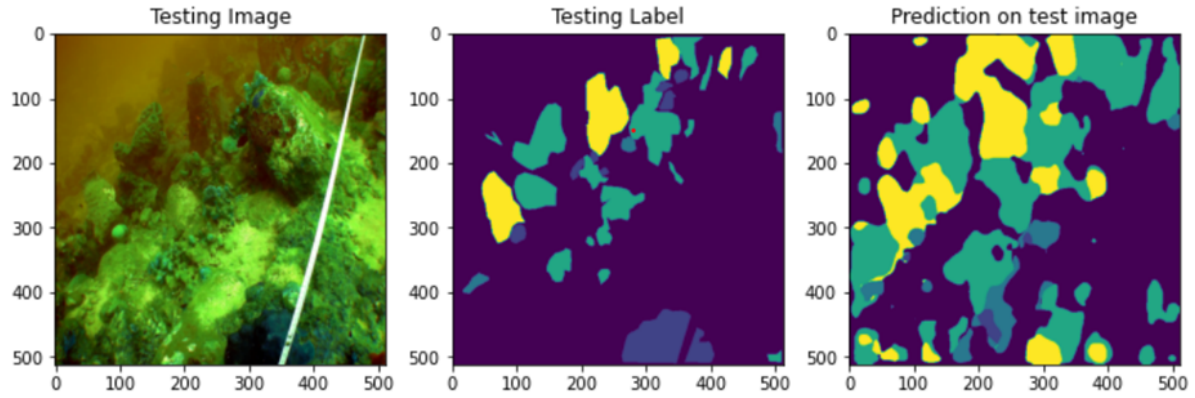

Figure 4.8: Prediction Without Image Filter

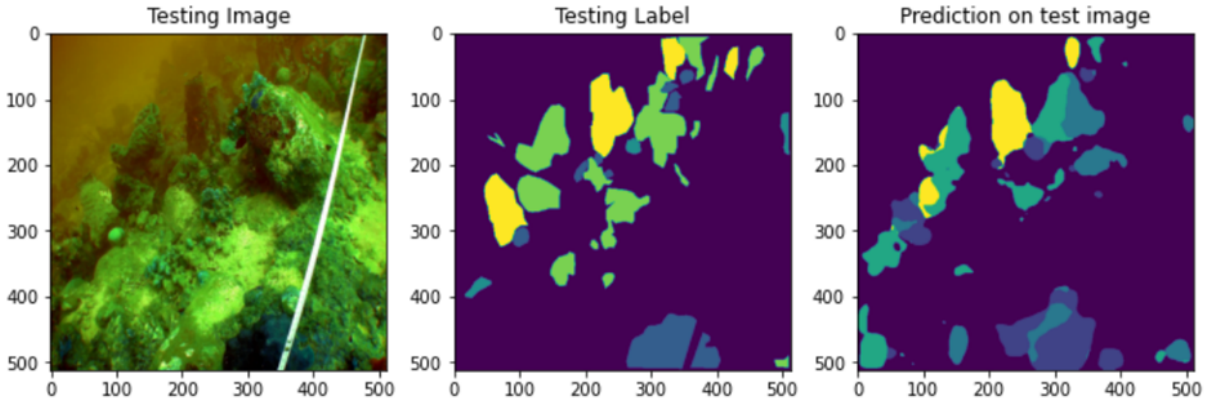

Figure 4.9: Prediction With Bilateral Filter

	Mean IoU
Bilateral Filter	0.234

Table 4.3: Results With Bilateral Filter

As we discussed before, we have created 6 sample datasets based on the location in which the images are collected. In the first 5 sample set each folder contains images from a single location and the sixth dataset is actually a mix of images equally taken from all the 5 sample sets. The model is trained on individual samples and predictions are made on the images from the same sample. Then the model is trained on the MIX dataset and predictions are made on the images from the same mix sample and also on the images from 5 individual sample sets. The mean IoU values are calculated against each and their values are compared at the end using a T-Test.

Sample dataset 1 : 20170803-dominica-cabrits

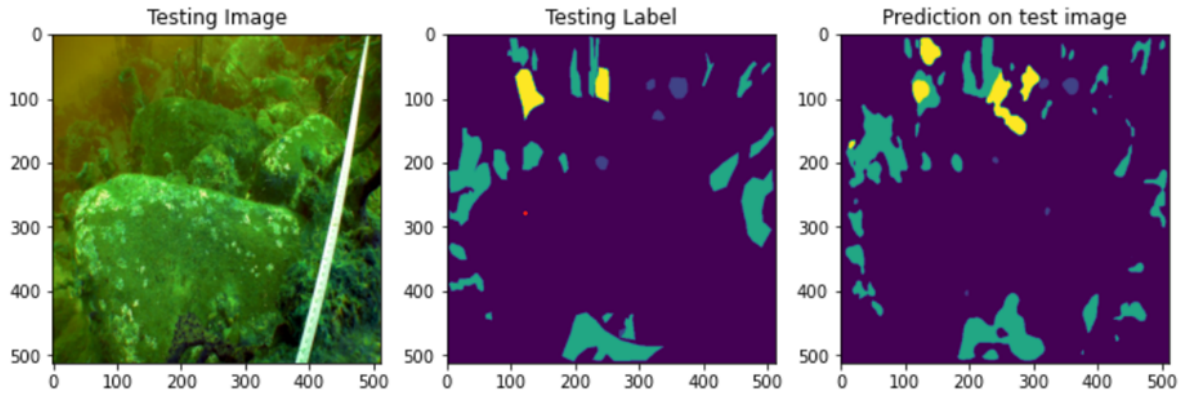

Figure 4.10: Sample dataset 1 : 20170803-dominica-cabrits

Sample dataset 2 : 20180406-spermonde-keke

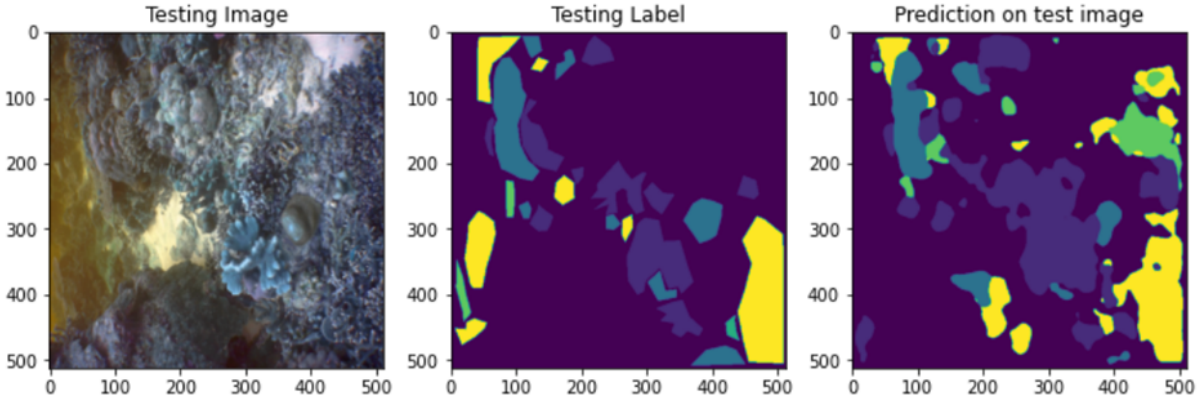

Figure 4.11: Sample dataset 2 : 20180406-spermonde-keke

Sample dataset 3 : 20190417-seychelles-BL

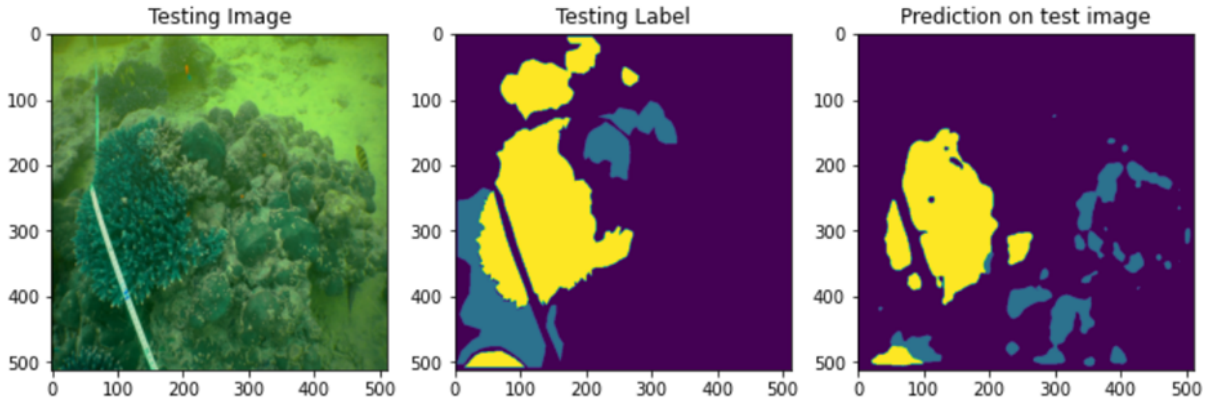

Figure 4.12: Sample dataset 3 : 20190417-seychelles-BL

Sample dataset 4 : K1-20180712-01

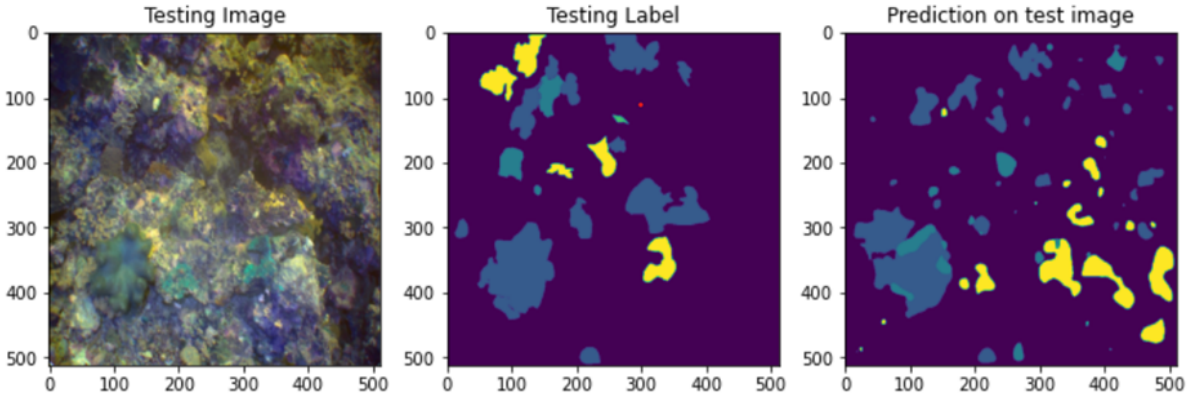

Figure 4.13: Sample dataset 4 : K1-20180712-01

Sample dataset 5 : PK-20180714-01

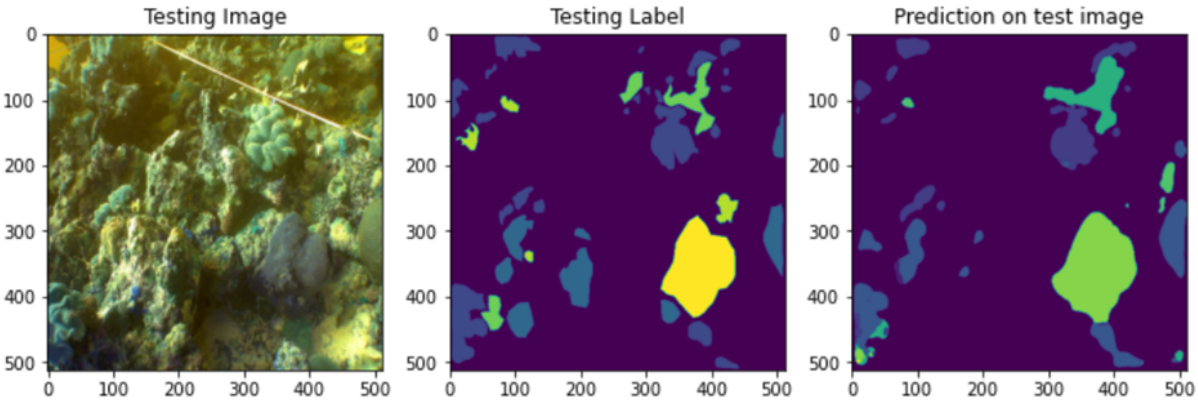

Figure 4.14: Sample dataset 5 : PK-20180714-01

Sample dataset 6 : MIX

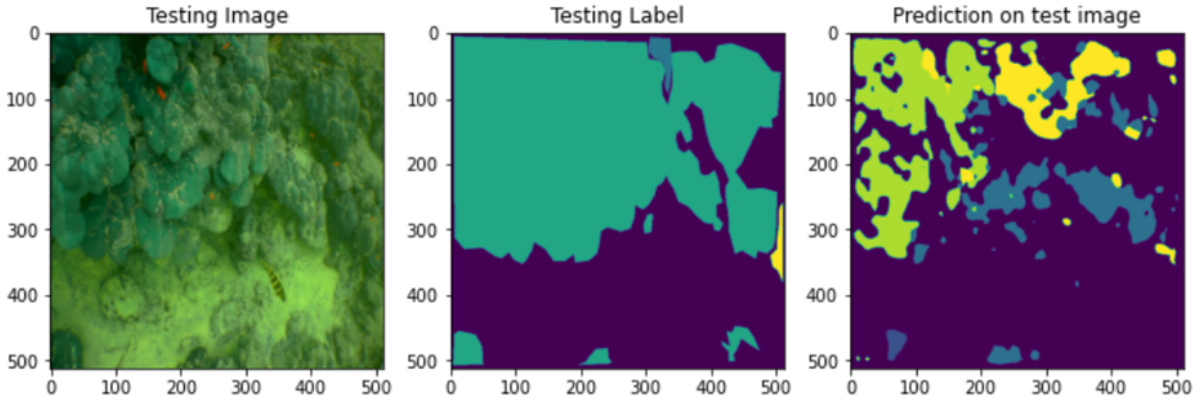

Figure 4.15: Sample dataset 6 : Prediction 1

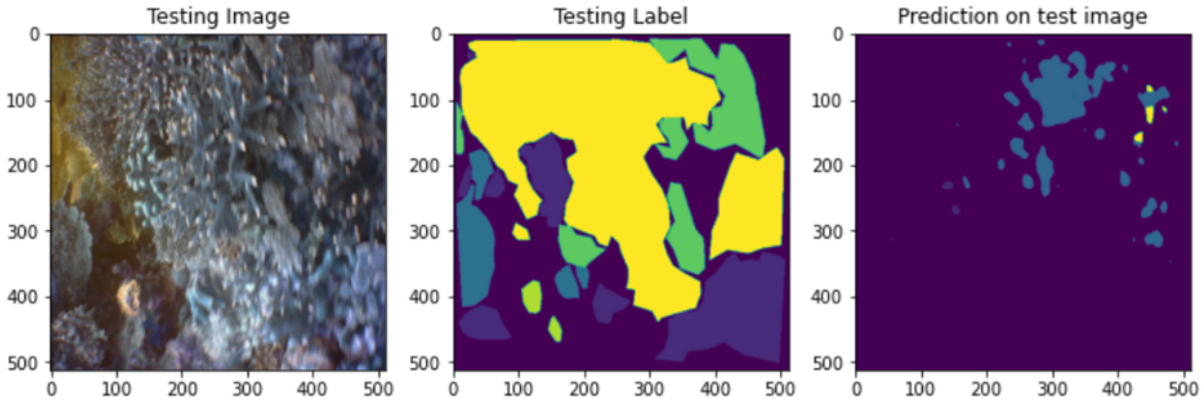

Figure 4.16: Sample dataset 6 : Prediction 2

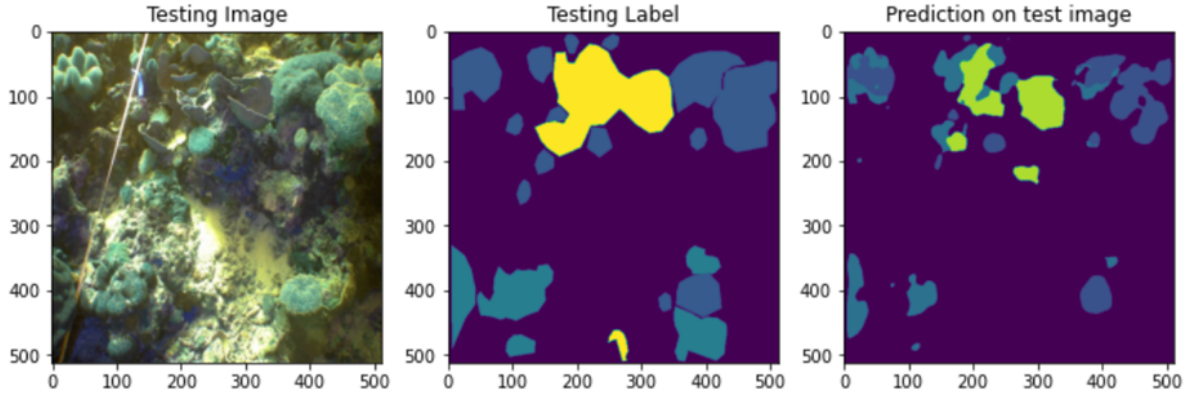

Figure 4.17: Sample dataset 6 : Prediction 3

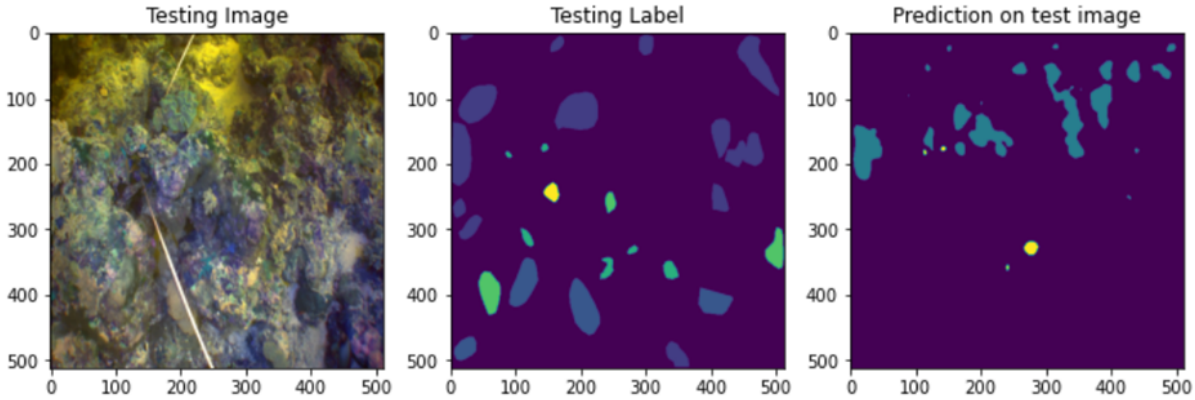

Figure 4.18: Sample dataset 6 : Prediction 4

	Mean IoU	Iou with MIX Model
20170803-dominica-cabrits	0.458	0.0979
20180406-spermonde-keke	0.2052	0.0557
20190417-seychelles-BL	0.284	0.1006
K1-20180712-01	0.149	0.067
PK-20180714-01	0.2056	0.068
MIX	0.104	NILL

Table 4.4: IoU values of sample datasets

T- Test

20190417-seychelles-BL	MIX	20170803-dominica-cabrits	MIX	20180406-spermonde-keke	MIX
0.28868842	0.09255439	0.4599088	0.13298364	0.23964605	0.05768632
0.32125458	0.09342734	0.4988481	0.11433758	0.2658609	0.05443966
0.31770828	0.08210443	0.49261162	0.09859553	0.31599587	0.052777942
0.29665288	0.095766574	0.48060355	0.10168566	0.19196168	0.051583488
0.28658205	0.10061954	0.451437	0.10101879	0.18058328	0.056389652
0.2906513	0.110370964	0.44945362	0.08898445	0.16506968	0.058404043
0.27768722	0.11109247	0.4626638	0.085978895	0.17177315	0.055883247
0.2848548	0.11075318	0.42839015	0.08468106	0.16855091	0.058129318
0.23809358	0.10663666	0.4283948	0.08389334	0.1776105	0.05705972
0.24067423	0.10274246	0.42776978	0.08691475	0.17591837	0.05504049
0.284284734	**0.100606801**	**0.458008122**	**0.09790737**	**0.205297039**	**0.055739388**
p-value	9.87341E-14	**p-value**	1.75398E-18	**p-value**	3.04907E-08

Figure 4.19: T-Test : Result

T - Test is used to compare the means of two groups, it will give the significance between the both and how they are related to each other. Here we chose first 3 sample datasets: sample 1, sample 2 and sample 3. The IoU values of 10 images from each sample is calculated and compared against the IoU values that we got with the model trained on MIX dataset. The T-Test is done on the mean values of the 10 images and the comparison is based on the p-values obtained.

T - Test is used to compare the means of two groups, it will give the significance between the both and how they are related to each other. Here we chose first 3 sample datasets: sample 1, sample 2 and sample 3. The IoU values of 10 images from each sample is calculated and compared against the IoU values that we got with the model trained on MIX dataset.

The T-Test is done on the mean values of the 10 images and the comparison is based on the p-values obtained.

We did the T-Test with null hypothesis, by considering the difference between the means of the two groups are zero. The test performed here is a two-tailed test with two sample equal variance and the result that we are evaluating here is the p-value with 99 percent confidence level.

- For the sample set one there is a significant difference in the mean IoU values of the model trained on 20190417-seychelles-BL (M = 0.284, SD = 0.0274) and the model trained on MIX dataset (M = 0.100, SD = 0.0096, alpha = 0.01); p = 9.87341E-14.

- For the sample set two there is a significant difference in the mean IoU values of the model trained on 20170803-dominica-cabrits (M = 0.458, SD = 0.0262) and the model trained on MIX dataset (M = 0.0979, SD = 0.015,$\alpha = 0.01); p = 1.75398E - 18.$

- For the sample set three there is a significant difference in the mean IoU values of the model trained on 20180406-spermonde-keke (M = 0.205, SD = 0.0512) and the model trained on MIX dataset (M = 0.0557, SD = 0.00228,$\alpha = 0.01); p = 3.04907E - 08.$

In all the three cases the null hypothesis is rejected and the alternate hypothesis is accepted.

CHAPTER 5

Discussions and Conclusions

This project was an attempt to answer the research questions that arise during the literature survey. We have tried different methods to find the best possible approach to get an optimum model performance. By visually analysing the figures 4.1, 4.2, and 4.3 the model trained at 5 and 10 epochs gives the best results and when we increase the epoch to 20 we got the least accurate prediction. Also, validation loss is minimum at 5 to 10 epochs and then gradually increasing. This is because of overfitting. we set the number of epochs at 10 for optimum performance. Normally when we have a small dataset we use different augmentation methods to create more train images and thereby improving the model performance. But here from table 4.1 we can see that the augmentation doesn't improve the model performance instead of that, it is negatively impacting the performance. Because the augmentation may makes the model difficult in finding a spurious patterns in our image data. The table 4.2 shows the mean IoU values of different loss functions. Even though the dice loss is very efficient for handling the class imbalance, categorical cross entropy had comparatively better result. Since we already used the Squeeze-and-Excitation Block (SENets) in our architecture to make it compatible with the class imbalance and the gradient is much nicer in categorical cross entropy compared to the dice loss. The maximum IoU value for the model is obtained by using an image filter in the train images [ref table 4.3] . The bilateral filter used here will remove the noises and smoothens the image by replacing the intensity values of each pixel compared to the weighted values of the neighbouring pixels. So in short good quality images with less noises can produce better results.

We also tried to find out whether it possible to came up with a single model for predicting the images from different locations. Our results shows that the models trained on individual locations gave us better results compared to the model trained on images from different locations. This is due to the fact that the images from different locations will have different texture, colour etc. Even the same substrate have different shape and colour variants in different locations. This will confuse the model while doing the prediction which results in poor performance. By analysing the table 4.4 the sample dataset of dominica-cabrits has the maximum results (0.458) followed by seychelles-BL (0.284) and spermonde-keke (0.205).The lowest result (0.149) was from the model trained on the sample set of K1-20180712-0. This variation in model performance is due to the number of substrate classes present in each sample set. Here the number of classes present in dominica-cabrits, seychelles-BL and spermonde-keke are 6, 8, and 9 respectively. So the performance of the model decreases when the number of classes increases.

The aim of the project was to do a detailed study of the previous editions and an attempt to find answers to the questions arise during the literature survey and we were successful in that. Due to the system limitations, we were not able to use the entire dataset but the approaches we followed mimicked the same environment and the results are comparable. The main difficulties we faced during the project was in the annotation task, since the raw data needed a lot of pre-processing in order to convert it into our required format. The performance of the model was directly proportional to the number of substrate classes present in the dataset also, it is very much dependable on the input image quality. The use of image filter in the train images really helped to improve the segmentation. By analysing table 3.4, the first 6 classes together consist the 91% of the total substrates and the last 5 classes has only 5% of the overall data. Even though we applied different methods to handle the class imbalance it is recommended to either neglect the classes with least no of substrate if possible or add more images into our dataset. An attempt was also tried to find out whether its possible to came up with a single model for predicting the images from different locations. By analysing our result, we suggest to create individual models against each locations. Since the model trained on the MIX dataset always underperformed the other models.

For the future studies we would recommend the following Methods/approaches that we think has an impact on the model performance:

- We resized the images to 512 x 512 in order to reduce the run time complexity, this

reduced the overall quality of the image. So images with higher resolutions can be tried and find out the impact on the results.

- Instead of randomly adding more images into the dataset which may increase the imbalance in the dataset, add more images in to the classes that has least no of substrates.
- Its is recommended to try different image filters and compare their impact on the prediction.
- Try the model with different backbones, here we used Resnet50 also, instead of deeplabv3+ other CNN architectures like U-net can be also used for the segmentation.
- Due to the system limitations, we trained the model with a batch size of one. Train the model with larger batch size like we did on the epochs and compare the results.
- Even though the augmentation didn't improve the model performance, still we could try image augmentation with different parameters and techniques.
- Instead of grouping the images based on the location, we could try to group them based on the substrate classes. This will helps to reduce the number of classes per dataset and may give better results.

  Overall, the results got for the base line model was moderate (sample set 1: dominica-cabrits : 0.152) compared to the previous editions. But after using different image enhancement methods and with optimum parameter values the performance of the model is increased near to good performance (sample set 1: dominica-cabrits : 0.458).

Bibliography

[1] Rachel Ross, September 24, 2018. "What Are Coral Reefs?" <https://www.livescience.com/40276-coral-reefs.html>

[2] NOAA, February 1, 2019. "Coral reef ecosystems". <https://www.noaa.gov/education/resource-collections/marine-life/coral-reef-ecosystems>

[3] Wright, Jessica P., Ioana-Lia Palosanu, Louis G. Clift, Alba GarcÃa Seco de Herrera, and Jon Chamberlain. "Pixelwise annotation of coral reef substrates." In CLEF (Working Notes), pp. 1394-1404. 2021.

[4] Bogomasov, Kirill, Philipp Grawe, and Stefan Conrad. "A two-staged Approach for Localization and Classification of Coral Reef Structures and Compositions." CLEF (Working Notes). 2019

[5] Steffens, A., Campello, A., Ravenscroft, J., Clark, A. F., Hagras, H. (2019). Deep Segmentation: using Deep Convolutional Networks for Coral Reef pixel-wise Parsing. In CLEF (Working Notes).

[6] 6. Soukup, Lukas. "Automatic Coral Reef Annotation, Localization and Pixel-wise Parsing Using Mask R-CNN." (2021)

[7] Nilesh Barla, PerceptronAI. October 21, 2022. "The Beginnerâs Guide to Semantic Segmentation" <https://www.v7labs.com/blog/semantic-segmentation-guide>

[8] Chen, Liang-Chieh, George Papandreou, Florian Schroff, and Hartwig Adam. "Re-thinking atrous convolution for semantic image segmentation." arXiv preprint arXiv:1706.05587 (2017).

[9] Paul-Louis Prove, Oct 17, 2017. "Squeeze-and-Excitation Networks" , <https://towardsdatascience.com/squeeze-and-excitation-networks-9ef5e71eacd7>

[10] prakharr0y, 24 Oct, 2020. "Intuition of Adam Optimizer" . <https://www.geeksforgeeks.org/intuition-of-adam-optimizer/>

[11] JEREMY JORDAN, 30 MAY 2018. "Evaluating image segmentation models". <https://www.jeremyjordan.me/evaluating-image-segmentation-models/>