HW2: Detecting underwater plants.

Javier Galindos (214373IV) Omar El Nahhas (214316IV)

29/12/2021

1 Introduction

The second task for the Computer Vision course (ITS8030) is to solve the problem of detecting underwater plants. Detection of vegetation in the sea is an important tool for the detection of the health of the water body[1]. The present project is developed using Python 3 and the AI-Lab hardware resources.

2 Methodology

2.1 Pre-processing

To have an estimation of the area covered for a specific underwater plant in the Baltic Sea, an object detection task is performed instead of classification. The species selected are Fucus, Furcellaria Lumbricalis and Zostera Marina. The unlabelled images have been hand-picked from the internet. Corners and edges, i.e. large shifts in pixel intensity, are what is mostly captured as features. An example of feature detection on an image with a watermark can be seen in Fig. 1. As a result, watermarks in an image need to be removed as the text contains all the characteristics of a good feature with its distinguishable appearance. The watermarks have been cropped and finally the images have been resized to a size of 1024x1024 pixels.

2.2 Etalons

All the etalons have been resized to 300x300 pixels for the sake of this report, although the original sizes are used for the feature detectors. The etalons for Fucus, Furcellaria Lumbricalis and Zostera Marina can be found in Fig. 2, Fig. 3 and Fig. 4 respectively. The etalons were chosen based on a human estimate of which features could describe/represent at least 50% of the images of that class. Three etalons per class were chosen, as that approximately covers atleast 50% of the sub-classes within the scope of the 50 images in each class. The colours themselves are irrelevant in this case, as the images are loaded in grayscale.

[1] http://stateofthebalticsea.helcom.fi/biodiversity-and-its-status/benthic-habitats/

Figure 1: Feature detection on Fucus with a watermark using ORB.

(a) (b) (c)

Figure 2: The chosen etalons for Fucus (resized 300x300 px).

2.3 Baseline

The baseline is defined as the best object detection result among ORB, SIFT and BRIEF models. In order to perform object detection using feature detectors, every image is split into a 4x4 grid, where every grid cell contains $\frac{1}{16}^{\text{th}}$ of the total image to be analysed. Based on the chosen feature detection models, the feature key points of all 9 etalons are calculated. Then, the chosen feature detector is applied on every cell of the training image. After all the feature descriptors and key points of both the etalons and the training image are known, it needs to be determined which features are seen as

(a) (b) (c)

Figure 3: The chosen etalons for Furcellaria Lumbricalis (resized 300x300 px).

(a) (b) (c)

Figure 4: The chosen etalons for Zostera Marina (resized 300x300 px).

"good" features. In order to determine how good a feature is, the L2 distance for floating points descriptors (i.e., SIFT) and Hamming distance for binary descriptors (i.e., ORB, BRIEF) between the query (etalon) key points and the training (image frame) key points are calculated. The smaller the distance, the better the features match. Using the 2 nearest neighbours, the Lowe's ratio test is applied to determine which key points to keep, and which to discard. The Lowe's ratio test is useful because it checks if the distances from the 2 nearest neighbours are sufficiently different. It works under the consumption that a **true** key point in the query image can only have 1 match in the training image frame, where the key point with the smallest distance is considered to be the match, and the key point with the second smallest distance is considered as random noise. As a result, if the matching key point is not different enough from the determined random noise, this matching key point is discarded and not taken in to account for further calculations.

As a result of filtering all the potential matches using Lowe's ratio, every image frame now has an amount of good matches from four classes: Fucus, Furcellaria Lumbricalis, Zostera Marina or None.

The "None" class is only assigned if there were zero found matches with the etalons of the other 3 classes. Per image frame, it is evaluated which type is dominant and is then labelled as such. If the amount of good matches of certain classes is equal, the image frame classification is split among those specific classes.

$$\begin{bmatrix} FucFurZos & Fuc & FurZos & FurZos \\ Fuc & FucFur & Fuc & Zos \\ None & Fuc & FurZos & Zos \\ FucFurZos & FucZos & Fur & FucFur \end{bmatrix}$$

After the 4x4 matrix is filled with the predictions from the feature detector, a 2x2 sliding window operation with a stride of 1 is performed on the matrix. This operation is done to get the most frequently occurring class within a 2x2 window and turn it into a classification where only 1 class exists. As the maximum probability in a 2x2 window is 4 (probability of 1 per grid cell), the occurrence of all the classes in the window are summed up, and the maximum class score is divided by 4 to get the overall confidence of this prediction. If the sum of occurence of 2 or more classes are the same, the prediction will be 'None' as there is too much confusion between classes. This sliding window operation leads to a 3x3 matrix.

$$\begin{bmatrix} Fuc & Fuc & Zos \\ Fuc & Fuc & Zos \\ Fuc & None & Fur \end{bmatrix} \xrightarrow[confidence]{} \begin{bmatrix} 0.708 & 0.625 & 0.5 \\ 0.625 & 0.625 & 0.625 \\ 0.458 & 0.375 & 0.5 \end{bmatrix}$$

After the sliding window operation, there is a 3x3 matrix with 9 grids which could result into 9 separate bounding boxes. Therefore, the flood fill algorithm is applied. This algorithm finds all connected grid cells in the matrix which share the same classification, essentially 'flooding' the imaginary boxes every class resides in. If there is only one occurrence of a certain class, it is considered as outlier and not counted for the final prediction.

$$\begin{bmatrix} Fuc & Fuc & Zos \\ Fuc & Fuc & Zos \\ Fuc & None & Fur \end{bmatrix} \xrightarrow[flood\ fill]{} \begin{bmatrix} Fuc & Fuc & Zos \\ Fuc & Fuc & Zos \\ Fuc & & \end{bmatrix}$$

Observing the matrix where the flood fill algorithm is applied, the connected Fucus grid cells are unable to form a rectangle in its current division. In this project, it is chosen to add extra grid cells to the shape in order to make a rectangle. If connected grid cells end up taking significantly more cells to form a rectangle, the confidence of the bigger rectangle is being punished. This is done by summing the confidence of all classes to which the rectangle will be classified to, and divide it by the amount of grid cells it occupies. As a result, the predicted final bounding boxes with accompanying confidence are created, which will be tested against the ground-truth bounding boxes for validation.

$$\begin{bmatrix} Fuc & Fuc & Zos \\ Fuc & Fuc & Zos \\ Fuc & \mathbf{Fuc} & \end{bmatrix} \xrightarrow[confidence]{} \begin{bmatrix} 0.708 & 0.625 & 0.5 \\ 0.625 & 0.625 & 0.625 \\ 0.458 & \mathbf{0} & \end{bmatrix} \xrightarrow[mean\ conf.]{} \begin{bmatrix} 0.507 & 0.563 \end{bmatrix}$$

2.4 Improving baseline: deep-learning

2.4.1 Detectron2

Towards improving the baseline results, a deep-learning approach is implemented. In particular, a Faster R-CNN on the Detectron2 framework is developed. The backbone of the Faster R-CNN net is RestNet-101-FPN. This DeepNet consist of four parts:

- A region proposal algorithm to generate "bounding boxes" or locations of possible objects in the image
- A feature generation stage to obtain features of these objects, usually using a CNN
- A classification layer to predict which class this object belongs to
- A regression layer to make the coordinates of the object bounding box more precise

The main difference between Faster R-CNN,R-CNN and Fast R-CNN are the region proposal algorithms. The Faster R-CNN fixes this by using another convolutional network (the Region Proposal Network (RPN)) to generate the region proposals instead of a selective search algorithm. This not only brings down the region proposal time from some seconds to some microseconds per image, but also allows the region proposal stage to share layers with the following detection stages, causing an overall improvement in feature representation.[2]

The net has been pre-trained on the COCO Dataset. For this project's implementation, the layers have been frozen during training except for the last classifier which is trained on our training set, thus performing transfer-learning.

2.5 Validation metrics

The feature detection and deep-learning models are validated using mean average precison (mAP), as it is an appropriate metric for the object detection task according to the theory. For this project specifically, the mAP used in the COCO challenge is selected, see COCO Evaluation. In this metric the Intersection over Union (IoU) thresholds vary from 0.5 to 1 in steps of 0.05. The mAP with an IoU threshold of 0.5 (PACAL VOC challenge) is also considered. Other metrics like accuracy, precision or recall by themselves could be biased and they do not present the real performance of the model.

[2]https://towardsdatascience.com/faster-r-cnn-for-object-detection-a-technical-summary-474c5b857b46

3 Results

The training dataset consists out of 165 images (57 of Fucus, 57 of Zostera Marina and 51 of Furcellaria Lumbricalis). Note, that the deep-learning model is trained with the whole training set while the feature detector model is "trained" with the etalons obtained from this training set. Towards having an unbiased validation of the model, a new dataset shared by fellow students is used for validation purposes. This dataset has not been cleaned or pre-processed. The reader should note that this dataset contains watermarks and is visually different (even for the human eye) from the training set. The validation set consists out of 104 images (58 of Fucus, 41 of Zostera Marina and 5 of Furcellaria Lumbricalis).

3.1 Feature detectors

Experiments have shown that a Lowe's ratio of 0.6 and a threshold of minimum good matches bigger than 0 leads to the most accurate results for the feature detectors within the scope of this project. Training and validation data in the space of feature detectors is not applicable, as the only "training" images that exist are the images where the etalons have been extracted from. For the sake of comparison to the Detectron2 results, the feature detectors results will still be divided into training and validation results. The results for the training dataset is seen in Table 1, and the result for the validation dataset is seen in Table 2.

Table 1: COCO and PASCAL VOC50 mAP metrics [%] per model on training dataset.

Model	**COCO**	**PV-50**
ORB	**3.00**	**7.10**
SIFT	2.90	7.10
BRIEF	2.20	5.10

Table 2: COCO and PASCAL VOC50 mAP metrics [%] per model on validation dataset.

Model	**COCO**	**PV-50**
ORB	**0.36**	**0.26**
SIFT	0.00	0.00
BRIEF	0.00	0.00

The performance of ORB and SIFT are performing similarly on the training dataset, while BRIEF is performing significantly worse. The performance on the validation set obtained from fellow students is significantly worse. Referring to the pre-processing section of this project regarding watermarks and its effect on feature detectors, the poor mAP on the validation set makes sense when observing a

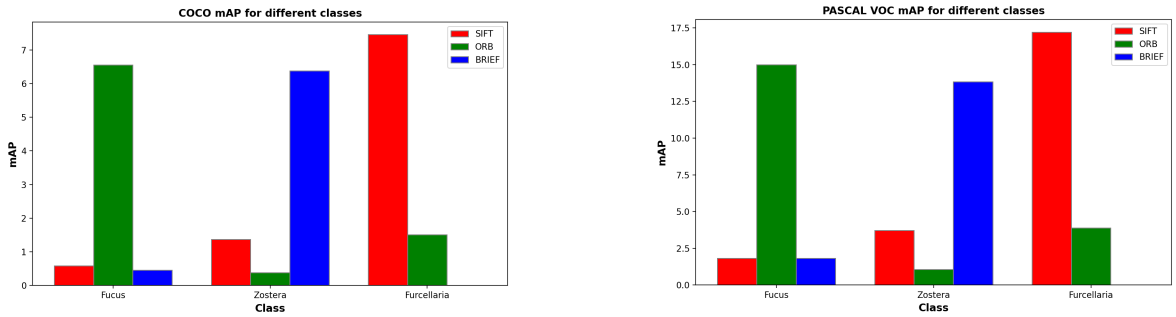
sample of the images in Fig. 5 which show many watermarks. This sample of the validation dataset is representative for the overall quality of the data from that set. Because of the poor validation results, this dataset is no longer considered valuable for further research within the scope of feature detectors unless the data is cleaned. Continuing with the results of the feature detectors on the training data, the performance of every model on every class is broken down in Table 3. Observing the results, an interesting pattern can be found in the "best performing class" per model. The ORB model performs best on Fucus, the SIFT model performs best on Zostera Marina and the BRIEF model performs best on Furcellaria Lumbricalis, while the same etalons are used for every model. This phenomenon is visualised in the bar plot in Fig. 6.

(a) (b) (c)

Figure 5: Sample of the raw validation set showing many watermarks.

Table 3: COCO and PASCAL VOC50 mAP metrics [%] per class on training dataset.

Model	**Class**	**COCO**	**PV-50**
ORB	**Fucus**	**6.56**	**15.00**
	Furcellaria	0.38	1.06
	Zostera	1.51	3.89
SIFT	Fucus	0.58	1.82
	Furcellaria	1.37	3.72
	Zostera	**7.46**	**17.22**
BRIEF	Fucus	0.45	1.82
	Furcellaria	**6.38**	**13.83**
	Zostera	0.00	0.00

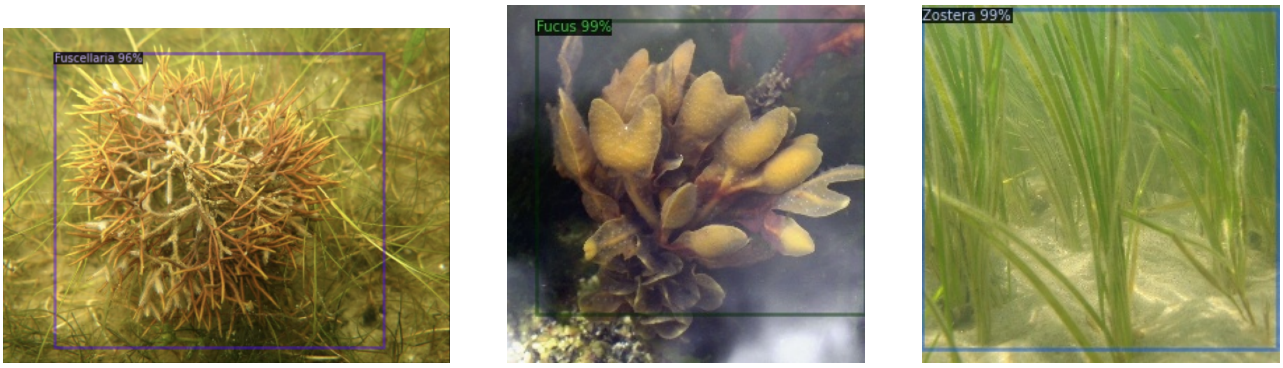


(a) COCO (b) PASCAL VOC

Figure 6: mAP metrics comparison for different classes and models on training dataset.

3.2 Deep-learning

The inference of the deep-learning model in the validation set is presented in Fig. 7.



(a) (b) (c)

Figure 7: Inference in the validation set using Detectron2.

The COCO mAP of the predictions on the **validation set** is **55.68%** and the PASCAL VOC mAP in the same set is **85.47%**. It is noteworthy to observe that the model is capable of detecting most of the sea species with an IoU threshold of 0.5 (PASCAL VOC mAP) but it slightly struggles for detecting the exact position and size of the plants (IoU 0.5 to 1). For this specific task, determining the correct position of the sea-plant is not critical, but predict the area where the plant is present is very important. Hence, both metric should be taken into consideration. Analyzing the COCO mAP metric per class (Table 4) shows that both Fucus and Zostera Marina are similarly detected. There are not enough samples of Furcellaria Lumbricais in the validation set to compute the mAP for this class which is comparable with the other classes.

The performance of the model is also analyzed in the **training set** to have more material for

[3]The sample size of this class is not comparable to the others (i.e., only 5 images)

Table 4: mAP metric per class on validation set.

Class	**COCO mAP**
Fucus	56.36
Zostera Marina	55.00
Furcellaria Lumbricalis	NA[3]

comparison with the baseline model. The COCO mAP is **82.58%** and the PASCAL VOC mAP is **99.66%** on the training set, which is naturally higher than the validation set mAP. In most cases, the algorithm correctly classifies the class and additionally determines the size of the specific plant with high precision. The model is capable of detecting the class Fucus slightly more accurate than the classes Zostera Marina and Furcellaria Lumbricalis as could be observed in Table 5.

Table 5: mAP metric per class on training set.

Class	**COCO mAP**
Fucus	86.31
Zostera Marina	80.83
Furcellaria Lumbricalis	80.59

The difference in performance between the training and validation set is summarised in Fig. 8. As one could expect, the performance is higher on the training according to both metrics. The difference is smaller when the IoU threshold is smaller (PASCAL VOC). Thus, the model performs better with unseen data when the localisation or size of the bounding box is less important. However, one should note that the validation set is not a representative sample of the training set. Relative to the training data, the validation data is visually significantly different and contains watermarks. It is also noteworthy that the size of the training set is relatively small ($\sim$ 50 images/class), hence the usage of the transfer learning approach. In traditional end-to-end learning, the size of the dataset is orders of magnitude bigger ($\sim 10^3 - 10^6\, images/class$).

4 Conclusion

The baseline mean-average precision (mAP) of the best feature detector on the given data and using 3 etalons per class, ORB, is COCO mAP 3% and PASCAL VOC50 mAP 7.1% on the training set, and COCO mAP 0.4% and PASCAL VOC50 mAP 0.3% on the validation set. The validation set contains many watermarks, and therefore the feature detector cannot be successfully applied to this data to detect seaweed species. The deep-learning approach using Detectron2, yields COCO mAP 82.6% and PASCAL VOC50 mAP 99.7% on the training set and COCO 55.7% and PASCAL VOC50

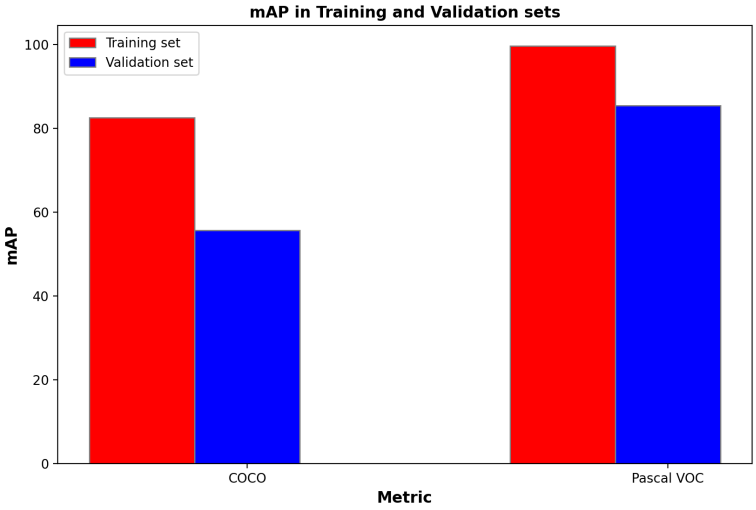


Figure 8: mAP metrics in Training and Validation sets.

mAP 85.5% on the validation set. Comparing the metrics on the training dataset, the baseline is improved 27x on the COCO metric and 14x on the PASCAL VOC50 mAP metric using Detectron2. For the validation dataset, which has a poor baseline because of the images with watermarks, the baseline is improved 139x on the COCO mAP metric and 285x on the PASCAL VOC50 mAP metric using Detectron2. Concluding, the deep-learning model significantly outperforms the baseline. The performance of the baseline is not acceptable for the task of performing object detection on seaweed species in the Baltic Sea. The deep-learning model satisfies the requirements and could be further implemented to perform the task.

5 Discussion

Improving the feature detector baseline could be done by finding better matching etalons, cleaner data, and utilising the observed phenomenon in the bias of different feature detector models on particular classes. The seaweed classes had many sub-classes within, which would require a significantly larger amount of etalons, but was limited to 3 per class for the sake of this project. In order to improve the reliability of the baseline improvement numbers, new validation data without watermarks should be generated containing an equal amount of samples across the classes. This was not possible in this research, as we have exhausted our online resources to find more images, and only one other group of our fellow students responded to our image exchange request which was used for the validation set in this research. Furthermore, further research should be conducted regarding the optimal way to handle the manual creation of bounding boxes and estimating their confidence.

The end-to-end approach of the deep-learning makes the development process substantially faster

and more straight-forward than the baseline due to readily existing libraries taking care of the processing pipeline. On the flip side, while the deep-learning model is a black box and it is not understandable why exactly the model is predicting a certain class, the predictions of the feature detectors could be understood with the manual selection of etalons, matches between etalons and images, etc. The phenomenon of overfitting on watermarks was quickly noticed during the "training" of the feature detectors, while noticing this could take significantly longer when training a deep-learning model because of the black box. Although the process of the baseline is more understandable than the deep-learning approach, the results show it clearly outperforms the baseline. To determine if a more explainable process is required for this use-case, further effort is required to research if the activation maps of the neural network are preferred over the ORB feature detection approach.

The deep-learning model could be improved by increasing the training sample size. If thousands of images are available instead of dozens, a neural net could be trained from scratch instead of applying transfer-learning with external weights. Thus, developing an end-to-end approach were the learning process (hyper-)parameters may be fine-tuned in more detail to this project's objectives.