Active Learning and object detection in aerial images for environmental studies

Master Thesis

For obtaining the degree

Master of Science (MSc)

at the Faculty of Natural Sciences of the Paris-Lodron-University Salzburg

Submitted by
Sarah Poch
12118710

Supervisors:

Assoc.-Prof. Dr Chloé Friguet
Université Bretagne Sud, IRISA-OBELIX

Dr Abdelbadie Belmouhcine
Postdoctoral researcher in IFREMER/IRISA-OBELIX

Dr Zahra Dabiri
Department of Geoinformatics

Salzburg, June 2023

Abstract

The combination of active learning and object detection is evaluated. On the object detection side, a detector from the well-know YOLO family, YOLOv7 is used. The active learning cycle is set as a pool-based algorithm, finding the most informative samples by uncertainty sampling. This framework is set up in the case of an environmental study, the detection of marine animals in aerial images. Moreover, the dataset has a hight *foreground-background* imbalance.

The results of this study shows that performance of object detection without active learning reaches at $mAP_{0.5}$ of 61% against 55% with active learning. But the integration of active learning shows other advantages. Among it, needing less non-empty images to reach this score.

Keywords - *Active Learning, Object Detection, YOLO, Machine Learning, environmental study*

Acknowledgement

I would first like to thank my UBS supervisors Dr. Chloé Friguet, Assoc.-Pr. and Dr. Abdelbadie Belmouhcine for their guidance and help through this master thesis.

I would also like to thank my PLUS supervisor Dr. Zahra Dabiri for her valuable comments on my thesis report.

I could not have undertaken this journey without my CDE Intake 3 fellow students and our weekly gathering meals.

Finally, I must thank my family and my friends for their continuous and unquestioning support throughout my years of studies and especially this master degree.

Contents

Acknowledgement **2**

List of Figures **5**

List of Tables **6**

1 Introduction **7**

2 Literature Review **8**
2.1 Active Learning 8
2.1.1 Active Learning Scenarios 8
2.1.2 Query strategies 10
2.2 Object Detection 11
2.2.1 YOLO object detectors 11
2.2.2 YOLOv7 12
2.3 Active Learning for Object Detection 13
2.3.1 Aggregation of the utility scores 13
2.3.2 Weighting utility scores 14
2.4 Imbalanced Data 14
2.5 Assessment 15

3 Application to an environmental study **17**
3.1 Dataset description 17
3.2 Exploratory Data Analysis 18

4 Methods **21**
4.1 Dataset 21
4.2 Active Learning for Object Detection 22

5 Experiments and results **23**
5.1 Experimental setup 23
5.2 Results and discussion 24
5.2.1 Results 24
5.2.2 Discussion 24

6 Conclusion **26**

Annexes **27**
A.1 Hyperparameters of YOLOv7 27

Glossary **29**

Bibliography **30**

List of Figures

2.1 Active learning pool-based cycle, adapted from [28] 9
2.2 YOLOv7 architecture for an input image of 640 x 640 pixels, from [40] 12
2.3 Confusion matrix obtained for a binary classification problem 15
2.4 Intersection-over-Union 16

3.1 Examples of patches belonging to the SEMMACAPE dataset of size 416x416 pixels. 18
3.2 Distribution of the objects per classes 19
3.3 Proportion and distribution of the images and objects in the SEMMACAPE dataset. 20
3.4 Bounding boxes surfaces. 20
3.5 K-means clustering of the bounding boxes width and height. Red points: cluster centroids. 20

5.1 Comparison of the proposed methods 25

List of Tables

4.1 Distribution of objects in the dataset used for running experiments 21

5.1 Distribution of L after the active learning cycles. 24

1 Hyperparameters tuning 28

Chapter 1

Introduction

In the current context, where more and more data are acquired and made available, the question of storage and resources needed to carry out analyses on this data arises. For example, the Copernicus Programme[1] satellite constellation alone acquires 20 TB of geodata every day. This represents a phenomenal amount of data to store, analyse and annotate. The relevance of using brute force, i.e., training the algorithm with all the data available for a project, is being called into question. In addition, the issue of better management of our energy resources is in the public eye. This means we need to find solutions that will enable us to manage the resources at our disposal better. Practices such as active learning, developed over several decades, are increasingly being considered to limit costs. Indeed, any geo-oriented data analysis can be costly in time and money. In the context of image analysis, it's time-consuming and fastidious to annotate Ground Truths on images, but it's also expensive to hire experts to carry out this task. The general idea behind active learning is to reduce the amount of data needed to carry out an analysis while simultaneously improving performance.

This project aims to assess whether active learning offers added value for object detection in environmental studies using aerial images. More specifically, as part of an environmental study aimed at detecting the presence of marine megafauna animals around wind farms located close to the coast in the Atlantic Ocean. The quantity of images available in this study is significant, but the presence of animals such as dolphins and birds remains a rare event.

For this project, we targeted 3 major research problems. First, the reduction of the amount of data needed to train a model and the selection of informative data by using active learning. Secondly, the use of highly imbalanced data, i.e., where the presence of animals is much lower than that of the ocean floor, and third, object detection. Although this is not the main focus, this aspect will be developed, but the project will not compare several object detection methods.

This report is organised as follows. First, Section 2.1 presents a brief description of the methodological and theoretical concepts underlying active learning and object detection. Then, in Section 3, the data from our environmental study are described and analysed. Next, the approach we used is described in Section 4. In Section 5, our experimental method is set up and tested in different configurations. Finally, Section 6 concludes our report.

[1] `www.copernicus.eu`

Chapter 2

Literature Review

2.1 Active Learning

Active Learning (AL) is a sub-category of Machine Learning (ML) and therefore by extension a sub-category of Artificial Intelligence (AI). The development of AL is part of an approach that requires less annotated data. Indeed, the goal is to choose the data with which the algorithm will learn most efficiently with the least amount of labelled data.

Rather than feeding on a lot of labelled data which can be costly (in time and money) and difficult to obtain (need of experts), AL tends to reduce the cost and amount of data needed to train a model correctly while still having a high level of accuracy. Also, reducing the amount of data used allows one to consider that each instance is not providing the same level of informativeness. AL with a wise selection strategy of instances associated with the model will tend to add a maximum of information for improving the training. In one line, an AL algorithm queries unlabelled data to select a particular instance that an oracle will annotate.

Different AL strategies exist at different stages of the algorithm workflow. Several scenarios have been developed for querying data. Later in the framework, it is possible to elect which instance is the more informative through some query strategies.

Some notation needs to be introduced and will keep the same meaning throughout this report. If needed, the notation will be defined again later. θ is the supervised model, Q is the query criterion and O is the oracle. In most cases, O is a human annotator. About the dataset, L is its labelled subset, and U is its unlabelled one. Inside these subsets, $\mathbf{x}$ is an instance and $\mathbf{x}^*$ a selected instance, and y a label belonging to Y, the set of all the possible labels. Depending on the problem, $\mathbf{x}$ can be a detection or an image. More formally, U and L are defined such as : $U = \{\mathbf{x}_1, ..., \mathbf{x}_i, ..., \mathbf{x}_m\}$ with $i \in [\![1, m]\!]$ and $L = \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_j, y_j), ..., (\mathbf{x}_n\, y_n)\}$ with $j \in [\![1, n]\!]$; $n = |L|$ and $m = |U|$.

2.1.1 Active Learning Scenarios

In the literature review of [28], three main AL strategies are highlighted, (i) pool-based AL, (ii) membership query synthesis, and (iii) stream-based selective sampling.

Pool-based AL

Pool-based active learning is cyclical, as illustrated in figure 2.1. More formally, as described by [11], the cycle proceeds as follows. Let θ trained incrementally with L, Q searches U by ranking

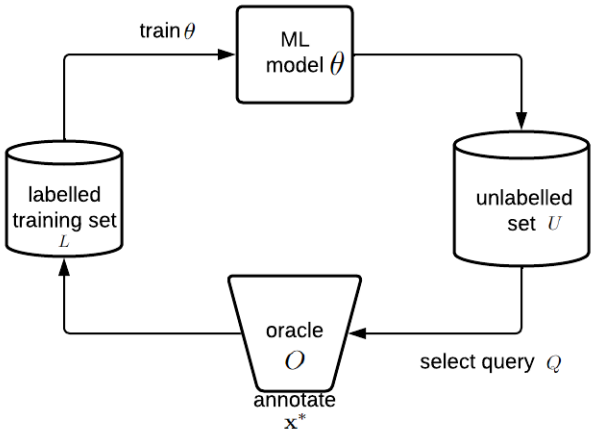


Figure 2.1: Active learning pool-based cycle, adapted from [28]

all instances of U and selects $\mathbf{x}^*$, and asks O to annotate it. This new annotated instance is added to L and removed from U. Then, θ is trained again with L, and the process is repeated until a stop condition is met.

In this scenario, L is assumed to be a small data set and U a large one. Moreover, U is, in most cases, assumed to be a finite set and available at once.

Membership query synthesis

This type of scenario was one of the first to be explored during the emergence of AL in the 1980s [1]. It involves the generation of new instances by the active learning model itself. The instance to be annotated is then selected from the entire starting space, including these newly created instances.

One of the major problems of this method is encountered when O, who has to annotate the instance, is a human. Indeed, instances created *de novo* may have no apparent meaning, even for an expert, as the object is artificially created. Nevertheless, this method is interesting in the case of a finite problem with a reduced data set. Membership query synthesis is commonly used in pool-based active learning, where a large pool of unlabelled instances is available.

Stream-based selective sampling

Unlike the membership query synthesis scenario, the selective sampling framework does not need to synthesise new samples. However, this technique shares the same backbone as the pool-based scenario. The difference is that selective sampling will sequentially evaluate each instance alone to decide whether to query it. In contrast, the pool-based technique will evaluate all the instances to rank them and pick the more interesting one to query.

The pool-based AL approach is the scenario which fits the best for object detection in environmental studies.

2.1.2 Query strategies

Now that a scenario is selected, it is time to select the more informative and useful samples to add to L. Only the uncertainty sampling query strategy is explained in this report. Other strategies such as Query-by-Committee, Expected Model Change, or Expected Error Reduction can be used [28].

Uncertainty Sampling

Uncertainty sampling is a successful and commonly used query method. This framework aims to select the unlabelled instances with a maximum uncertainty rate [28]. The main hypothesis is the more uncertain the selected unlabelled instance is, the more informative it will be. In a general case of classification, the workflow is described as follows and developed in the Algorithm 1 adapted from [23]. Let $b \in \mathbb{N}$, the number of unlabelled instances to be queried. The utility score, $s(\theta, \mathbf{x}_j)$ with $\mathbf{x}_j \in U$, is calculated for each instance of U and $\mathbf{x}^*$ is the optimal instance that will be annotated. The utility score characterises the uncertainty of an instance. Measures like entropy, least confidence, or margin sampling can be used to compute this score.

Algorithm 1 Uncertainty Sampling

```
initialise i = 0
while i < b do
  for all x ∈ U do
    calculate s(θ, x)
  end for
  annotate x*
  L = L ∪ {(x*, y*)}
  U = U\{x*}
  train θ with L
  i = i + 1
end while
```

The utility score is obtained by using different methods. The most known ones are the least confidence, margin sampling, and entropy.

Least confidence Least confidence uncertainty measure is calculated as :

$$s_{LC}(\theta, \mathbf{x}) = 1 - p_\theta(y_\alpha|\mathbf{x}) \tag{2.1}$$

where $y_\alpha = \underset{y \in Y}{\operatorname{argmax}}\ p_\theta(y|\mathbf{x})$, the class with the highest probability score for the model θ. In this case, $\mathbf{x}^*_{LC}$ will be :

$$\mathbf{x}^*_{LC} = \underset{x \in U}{\operatorname{argmax}}\ s_{LC}(\theta, \mathbf{x}) \tag{2.2}$$

Margin Sampling Instead of the least confidence measure, the margin sampling uncertainty measure considers more than one label probability. The two highest probable classes define this measure, as explained in [28].

$$s_{MS}(\theta, \mathbf{x}) = p_\theta(y_\alpha|\mathbf{x}) - p_\theta(y_\beta|\mathbf{x}) \text{ and } \mathbf{x}^*_{MS} = \underset{x \in U}{\operatorname{argmin}}\ s_{MS}(\theta, \mathbf{x}) \tag{2.3}$$

where $y_\alpha = \underset{y \in Y}{\operatorname{argmax}}\ p_\theta(y|\mathbf{x})$ and $y_\beta = \underset{y \in Y \setminus y_\alpha}{\operatorname{argmax}}\ p_\theta(y|\mathbf{x})$, the two first classes with the highest probability score for the model θ. As $s_{MS}(\theta, \mathbf{x}) \in [0, 1]$, it is also possible to use : $\mathbf{x}^*_{MS} = \underset{x \in U}{\operatorname{argmax}}\ 1 - s_{MS}(\theta, \mathbf{x})$.

Differentiating instances with large margins is easier for the model than differentiating instances with small margins, i.e. when the probabilities for the two first classes are close to each other. Thus, it is more interesting for the model to obtain the label of an instance in the latter case.

Entropy Still, according to [28], the entropy measure is the more commonly used method and a better generalisation. Unlike the margin sampling or the least confidence uncertainty measure, the calculus of entropy take into account all the possible class probabilities.

$$s_E(\theta, \mathbf{x}) = -\sum_{y \in Y} p_\theta(y|\mathbf{x}) \log(p_\theta(y|\mathbf{x})) \text{ and } \mathbf{x}^*_E = \underset{x \in U}{\operatorname{argmax}}\ s_E(\theta, \mathbf{x}) \tag{2.4}$$

In the case of a binary classification where $Y = \{0; 1\}$, e.g. this study, and for the three mentioned measurements, the most interesting instances have a probability score close to 0.5. In such a setting, the difference between $p_\theta(0|\mathbf{x})$ and $p_\theta(1|\mathbf{x})$ will tend towards 0 [23].

For the following experiments, the AL framework is a pool-based scenario with the uncertainty sampling query method. All three utility score computation methods are proportional in the case of binary classification; hence we can choose anyone. In our experiments, the utility score of each instance is computed using the margin sampling measure.

2.2 Object Detection

Comprehending the nature and location of objects in an image helps one understand the whole image. This task is object detection [39]. Object detection can be used on images and videos and has multiple applications, e.g. face recognition, text detection and many more. This task belongs to the computer vision and deep learning tasks and uses mainly Neural Network architectures such as Convolutional Neural Networks (CNN). The aim of Object Detection (OD) is both locating and classifying objects in an image. These detected objects are delineated by a bounding box and labelled. The framework of object detection, according to [39], has three parts; Informative Region Selection, Feature Extraction and Classification.

Two main algorithm families can be identified; two-stage detectors, i.e., Region-based methods including Region based Convolutional Neural Networks (R-CNN), Fast R-CNN and Faster R-CNN and one-stage detectors, i.e., Regression/Classification based methods [39]. These latter include the You Only Look Once (YOLO) object detector family.

2.2.1 YOLO object detectors

YOLO is a one-stage detector that considers detection as a regression problem. Both detection and classification are done at the same time. It uses a CNN architecture, and the processing is faster than the processing of two-stage detectors. This is justified by the usage of the simplest components but also by the fact that the model is run only one time on each image. That is the reason behind the name "You Only Look Once". This family of detectors is predominant in the field of real-time object detectors.

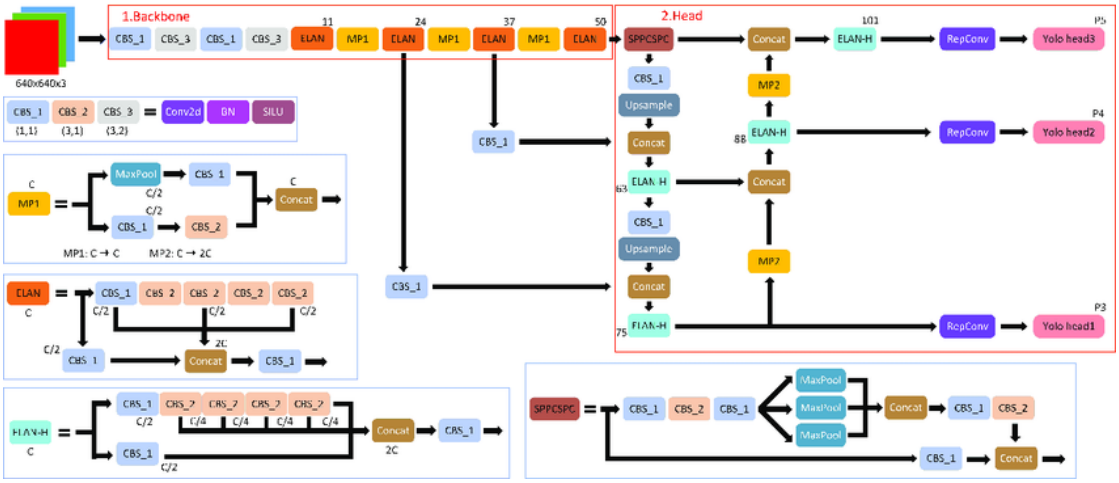


Figure 2.2: YOLOv7 architecture for an input image of 640 x 640 pixels, from [40]

The main workflow of YOLO models is as follows. Each image is passed through the network to obtain a grid of size S x S. Each cell of the grid predicts an offset with respect to A_S standardised bounding boxes. These fixed sizes bounding boxes are also called anchor boxes. Their size is specific for each dataset and needs to be calculated according to the size of the objects already annotated. For each grid cell, the output is a tensor with a shape of A_S x $(5 + n_c)$, with n_c the number of classes in the dataset. From YOLOv3, the detections are made on different scales. The network does not provide one grid per image but several grids per image. On each scale, the image produces a cell grid of size νS x νS where $\nu \in [\![1, v]\!]$ - v the number of scales.

In our case, each image has three scales. The three grid cells have a size of respectively 8 x 8, 16 x 16 and 32 x 32 pixels. Thanks to the k-means clustering that will be presented in Section 3, three anchor boxes are defined with the following sizes: $a_1 = (52, 94), a_2 = (80, 55)$ and $a_3 = (29, 30)$.

2.2.2 YOLOv7

YOLOv7 is the latest stable version of YOLO. According to [35], this new version outperforms the speed and accuracy of the other real-time object detectors. YOLOv7 shares its architecture with older versions of the detector, among them YOLOv4 with some modifications. As a brief description of YOLOv7 architecture, it contains a backbone, a head and a neck as shown in Figure 2.2. In the paper [35], the authors introduce the utilisation of an Extended Efficient Layer Aggregation Network (E-ELAN). This module is pictured in the above mentionned figure. The module Cross-Stage Partial Fast Spatial Pyramid Pooling (SPPCSPC) is also introduiced and presented as a more efficient method than a simple max pooling process.

2.3 Active Learning for Object Detection

The usage of AL is already widely spread for classification problems. In the case of OD, there is still room for improvement. The authors of [6] proposed a method which adopts classification using AL for OD.

In the case of classification tasks, a single prediction is selected and then annotated. However, the rest of the image is not considered and annotated. [6] poses the following hypothesis: the most informative samples tend to be found within the same image. The proposed solution is, therefore, to select and annotate an entire image rather than a single prediction. In this way, all the predictions previously made on the image will be validated and corrected by the annotator.

The aim is to find the most informative image. To achieve this, the uncertainty sampling query strategy is used. This strategy has been explained in Section 2.1. The first step is to calculate the utility score $s(\theta, \mathbf{x})$ for all samples belonging to U. This score is calculated with the margin sampling method. The formula is defined slightly differently from [28], but the result remains equivalent.

$$s_{MS}(\theta, \mathbf{x}) = (1 - (\max_{y_\alpha \in Y} p_\theta(y_\alpha|\mathbf{x}) - \max_{y_\beta \in Y \setminus y_\alpha} p_\theta(y_\beta|\mathbf{x})))^2 \tag{2.5}$$

Then, scores are aggregated per image, and the image with the highest score is fully annotated. Different aggregation methods can be applied.

2.3.1 Aggregation of the utility scores

Different methods are available to aggregate the utility scores per image. Among them, we have summing, averaging or finding the maximum value. In each case, it is also important to define the utility score for an image not containing any detection. For the three methods presented, this type of image has a value of zero. A new notation is introduced, D is the number of detection associated with an image. This number is variable from one image to another.

Sum One way to aggregate utility scores is to sum it for each image, such as :

$$s_{Sum}(\theta, \mathbf{x}) = \sum_{i \in D} s_{MS}(\theta, \mathbf{x}_i) \tag{2.6}$$

The limit of this aggregation method is that the sum is sensitive to D. Intuitively, an image with many detections will be selected faster than one with only a few.

Average To avoid this sensitivity, the calculation of the average is proposed. Dividing the sum by D normalises the global score. This score is within the range of 0 and 1. The comparison between images is made easier.

$$s_{Avg}(\theta, \mathbf{x}) = \frac{1}{|D|} \sum_{i \in D} s_{MS}(\theta, \mathbf{x}_i) \tag{2.7}$$

Maximum Lastly, finding the maximum among all individual detections is another way to aggregate results.

$$s_{Max}(\theta, \mathbf{x}) = \max_{i \in D} s_{MS}(\theta, \mathbf{x}_i) \tag{2.8}$$

2.3.2 Weighting utility scores

In the case of class imbalance, selecting instances from different classes is important to avoid underrepresented classes in L. To overcome this problem, the utility score can be weighted by w_c defined as :

$$w_c = \frac{n_{instances} + n_{classes}}{n_{instances_c} + 1} \tag{2.9}$$

where c is the predicted class, $n_{instances}$ the number of instances in the training set and $n_{instances_c}$ the number of instances belonging to the class c in the training set. s_{MS} is multiplied by w_c before the aggregation of the utility scores.

2.4 Imbalanced Data

Several problems can be encountered in information extraction using a constantly increasing amount of data. The most common issues are how to store and process such large amounts of data and the resources needed to run any kind of analysis. However, another problem is related to the dataset's composition. This is the problem of imbalanced data, and it should be considered when studying the dataset. Indeed, imbalanced data can affect the quality and performance of most learning algorithms.

The study carried out in this thesis aims to detect the presence of animals in aerial images using active learning. This classification problem is a binary problem without considering the species of the detected animal. Therefore, only the imbalance between the target class and the background will be considered and explained here.

Imbalance problems can occur in several ways; for example, [24] identifies four main categories: class imbalance, scale imbalance, spatial imbalance, and objective imbalance. In this report, only the class imbalance, which is encountered by our dataset, is taken into account. This imbalance problem occurs when the data distribution between classes is highly unequal. Indeed, in such a dataset, one class will be more widely represented than the other.

The class imbalance is divided into two sub-categories. The *foreground-foreground* imbalance and the *background-foreground* imbalance [24]. The first one is encountered when one or several classes significantly dominate the other classes. The latter happens when the background is highly predominant with respect to the positive objects. Our dataset, composed of a predominant background part, is in this second sub-category, with only a few foreground objects.

Focal Loss One of the loss functions used to train a one-stage object detector is Focal Loss (FL). Furthermore, according to [20] and [24], this loss function is particularly suitable in the case of a *foreground-background* imbalance. In the case of binary classification, FL is defined as :

$$FL(p_t) = -(1 - p_t)^{\gamma} log(p_t) \tag{2.10}$$

with $p_t = \begin{cases} p & \text{if } y = 1 \\ 1 - p & \text{otherwise} \end{cases}$ and γ is a tunable parameter. This formula comes from the Cross Entropy (CE) loss function defined as :

$$CE(p_t) = -log(p_t) \tag{2.11}$$

In this study, the coefficient γ is set to 0, which means only the CE loss function is used.

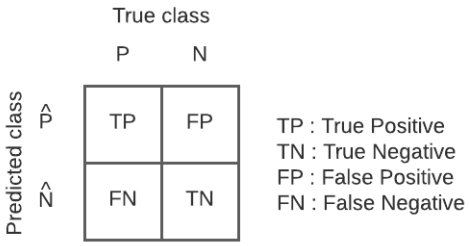


Figure 2.3: Confusion matrix obtained for a binary classification problem

2.5 Assessment

As explained in the previous sections, setting up the right metric according to the dataset and analysis is necessary. The evaluation of the performances needs to be adapted to the class imbalance problem, object detection and active learning methods.

All metrics explained in this section can be obtained from the confusion matrix shown in figure 2.3. The latter is easy to obtain and interpret in the binary classification framework. It will serve as a support and reference in this section. According to the notations defined on 2.3, $\{\hat{P}, \hat{N}\}$ are the predicted classes and $\{P, N\}$ the true classes.

Accuracy and Error Rate

The most commonly used assessment method is the calculation of the *accuracy* and the resulting *error rate*. The *accuracy* and the *error rate* are defined as follows:

$$accuracy = \frac{TP + TN}{TP + FN + TN + FP}, error\_rate = 1 - accuracy \tag{2.12}$$

In formal words, *accuracy* measures the ratio between the correct predictions and the total number of objects evaluated. Conversely, *error rate* gives the percentage of misclassified objects [21].

However, according to [14], these metrics are not adapted for evaluating problems with imbalanced data. Indeed, the calculation of *accuracy* is too sensitive to the data distribution. Precision, recall, F-score, or Area Under the Curve (AUC)-Precision-Recall (PR) are more relevant for the performance assessment of imbalanced learning algorithms. On the other hand, the currently used metric for object detection tasks is the mean Average Precision (mAP) [26]. This measure combines the previously mentioned metrics and does not require thresholding of confidence scores.

Precision and Recall

The *precision* and the *recall* are two other easy-to-compute metrics. The *precision* is the ratio between the number of correctly predicted positive objects TP and the number of predicted positive objects. The *recall* is the one between TP and the number of true positive objects [21]. These two metrics are defined as :

$$precision = \frac{TP}{TP + FP}, recall = \frac{TP}{TP + FN} \tag{2.13}$$

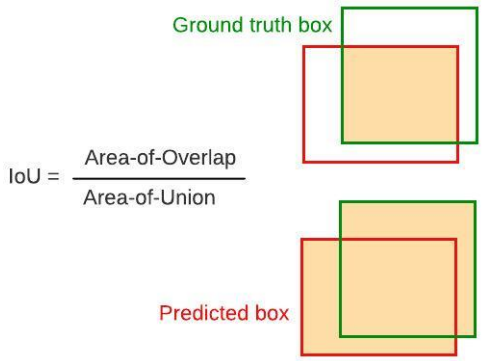


Figure 2.4: Intersection-over-Union

Intersection-over-Union (IoU)

The IoU indicator is usually used to evaluate object detection models like YOLO [4] or Fast R-CNN [25]. This metric calculates the overlapping between the predicted bounding box coordinates and the ground truth bounding box coordinates. The IoU is calculated as indicated in the figure 2.4. The value obtained is included in [0, 1]. The lower the IoU is the more the ground truth and predicted bounding boxes coordinates are different.

AUC-PR

The AUC-PR is a method used to visualise the algorithm's performance. These curves are also commonly used to compare the performances of different models [21]. The AUC-PR curve is obtained by plotting the *precision* over the *recall*, indicators defined earlier [14].

mAP

Now that *precision*, *recall*, IoU and AUC-PR are defined, it is possible to combine them to obtain a new metric, mAP. This metric also compares the Ground Truth and the detected bounding boxes. The *precision* and the *recall* are calculated according to the IoU values. A threshold IoU_{th} is applied to the IoU. The object detected with a IoU superior or equal to IoU_{th} is counted as a TP. On the contrary, if IoU is inferior to IoU_{th} it is counted as a FP.

The AUC-PR is displayed according to these $precision_{IoU_{th}}$ and $recall_{IoU_{th}}$. The Average Precision (AP) is calculated as the area under the curve of the PR curve. This score is included in [0, 1]. Finally, the mAP is the average of the AP scores over all classes. The higher the mAP score is, the more accurate the model is.

For the following experiments, the IoU_{th} is set to 0.5 (50%). The mAP_0.5 will be returned to show the model's performances.

Chapter 3

Application to an environmental study

3.1 Dataset description

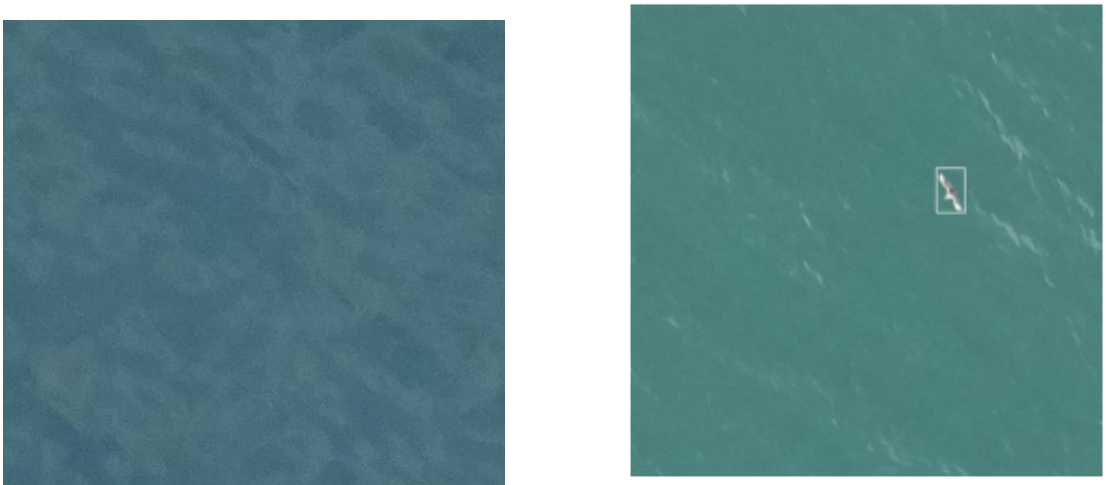
Aerial images used to conduct this study have been taken along the French coast and gathered via the monitoring and study of marine megafauna by automatic characterisation in wind farms (Suivi et Étude de la Mégafaune Marine par Caractérisation Automatique dans les Parcs Éoliens) (SEMMACAPE)[1] project. The results presented in this thesis are independent of this project. However, the data used were graciously lent to us by the University of Southern Brittany (UBS)[2]https://www.univ-ubs.fr and the Institute for Research in Computer Science and Random Systems (IRISA)[3], partners in the above-mentioned project.

Briefly, the SEMMACAPE project is a project carried out from 2019 to 2022 in collaboration with the UBS, IRISA, WIPSEA[4], France Energies Marines[5] and the French Office for Biodiversity (OFB)[6]. This project is in response to the current trend toward the development of marine renewable energies, particularly through the installation of offshore wind farms.

The acquisition of the images took place in 2020 and was carried out by l'Avion Jaune[7], a company subcontracted for aerial image acquisition. These images were taken by two cameras (Left and Right) and provided in JPEG format. In addition, these data are geo-referenced with the following parameters; GPS + roll, pitch, and yaw of the aircraft and the camera. The acquisition site is located in the South-West of France (Gironde Estuary and Pertuis Sea Natural Marine Park). Seven flights were conducted in 2020 to collect 92 463 images over a total area of 3 700 km^2. The resolution is 2cm x 2cm per pixel. Each image has a size of 14 204 x 10 652 pixels.

The SEMMACAPE dataset is a subset of the data described above. It contains 165 labelled aerial images of size 14 204 x 10 652 pixels. Each image is divided into smaller patches of size 416 x 416 pixels, as displayed in Figure 3.1. The dataset contains ground truth labels validated by a

[1] http://semmacape.irisa.fr/
[2] \unskip\protect\penalty\@M\vrulewidth\z@height\z@depth\dpff
[3] https://www.irisa.fr
[4] https://wipsea.fr
[5] https://www.france-energies-marines.org/
[6] https://www.ofb.gouv.fr/
[7] https://lavionjaune.com/



(a) Empty patch

(b) Patch with a flying little shearwater delineated

Figure 3.1: Examples of patches belonging to the SEMMACAPE dataset of size 416x416 pixels.

marine mega-fauna specialist. All this preprocessing was done in the SEMMACAPE project and is described in [3]. The data available are, therefore, 127 684 patches of size 416 x 416 pixels. As we are using images already preprocessed in the previous project, the term image will be used here to describe a patch as defined above.

Each non-empty image is associated with a text annotation file. An non-empty image is an image containing at least one objet, delineated by a bounding box. For the image in Figure 3.1b, the file contains the following information: ***[14 0.6779 0.3942 0.0601 0.0938 100]***. The annotation file provides information about the number of objects in the image and the bounding box coordinates containing the object of interest. The data correspond respectively to the class of the object, the coordinates of the centre of the box, (x, y), the width of the box, w and the height, h and p the probability to belong to the class. x, y, h and w are normalised. It is, therefore, necessary to multiply these values by 416 to find the box's coordinates on the image. The drawing of the box with the coordinates is done as shown in Figure 3.1b. This is the standard format for annotation files obtained with the YOLO object detector family [9]. Finally, each empty image will be associated with an empty annotation file.

3.2 Exploratory Data Analysis

Among the 127 684 images available, 1 707 contain at least one object, and the remaining 125 977 images are considered to be empty and contain background. The background is ocean waters, and its variation. Object are everything which is not water (marine life, boats, waste, etc). The percentage of non-empty images is 1.3%. Figure 3.3a illustrates this distribution and demonstrates an obvious imbalance between the presence of objects and the background, which we will refer to as *foreground-background* class imbalance [24]. The consequences of this imbalance are a necessary adaptation of the techniques and metrics to be used for object detection and active learning, discussed in section 2.4.

The objects are classified according to 20 classes, the nomenclature is visible in the Figure 3.2.

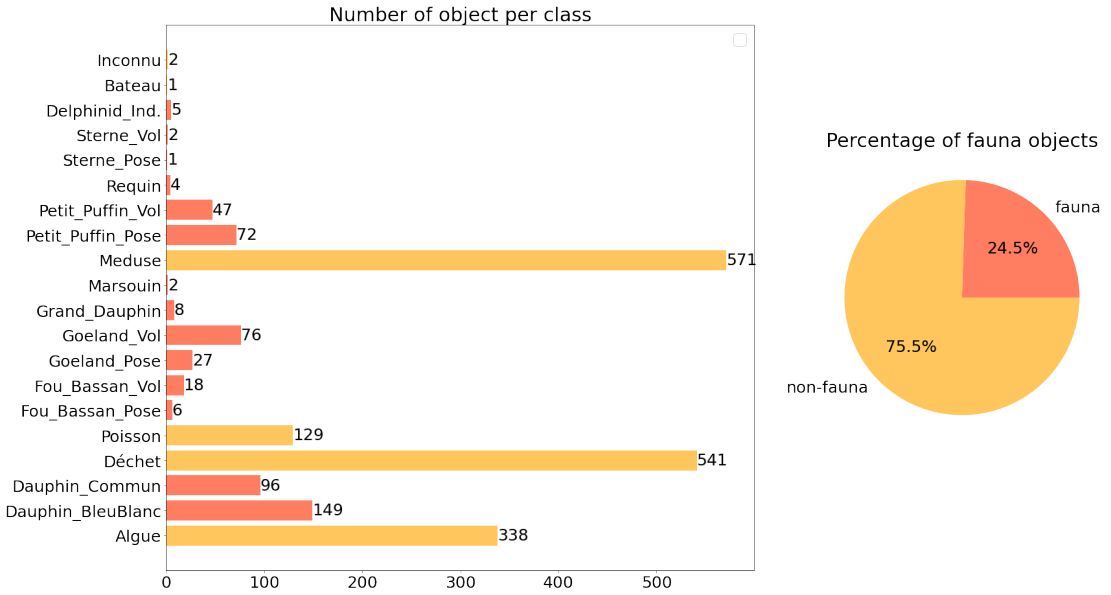


Figure 3.2: Distribution of the objects per classes

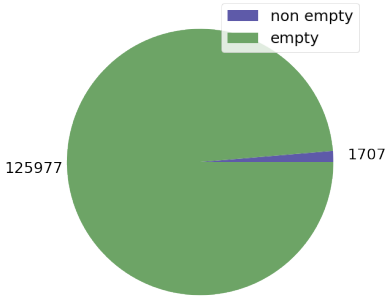
As mentionned earlier, the purpose of the project is to detect marine megafauna. In this respect, only the objects indentified as birds (terns, gannets, geoland, little shearwater), sharks, marine mammals (different species of dolphins) and cetaceans (porpoises) need to be detected. These classes will be grouped together in a single class called *fauna*. The objects not belonging to this reduced nomenclature ('Algue' (algae), 'Déchet' (waste), 'Inconnu' (unknown),'Bateau' (boat), 'Poisson' (fish) and 'Méduse' (jellyfish)) will be considered as background. The new *fauna* class represents 24.5% (513) of the 2095 objects in our dataset.

According to Figure 3.3b, in the 348 images containing at least one object *fauna*, 52.4% of the images contain only one object. In addition, only 5 images contain 6 or more objects belonging to this class. The number of objects per image distribution is highly heterogeneous.

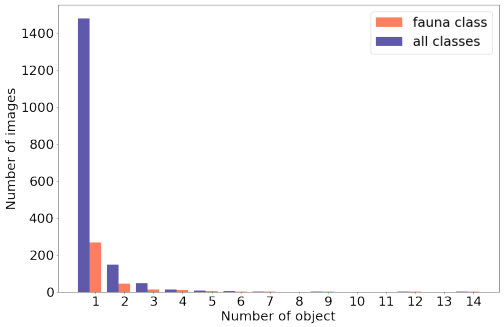
The study of the dimensions and surface of each object is also a valuable source of information for the next steps. Figure 3.4 displays the boxplot of bounding box surfaces for the *fauna* objects. These areas were obtained thanks to the annotation files mentioned above. The objects of interest have an average width of 49 pixels and an average height of 52 pixels. All our objects belonging to the class *fauna* have a height belonging to [10; 169] pixels and a width in [8; 147] pixels.

A k-means clustering analysis is led on the width w and the height h of the bounding boxes [22]. It is performed with different values of k, between 1 and 4. Figure 3.5 displays the cluster with the different values of k. The red points are the centroids and have the following coordinates :

- for $k = 1$, $c = (49, 53)$
- for $k = 2$, $c = [(36, 32), (68, 81)]$
- for $k = 3$, $c = [(52, 94), (80, 55), (29, 30)]$
- for $k = 4$, $c = [(57, 44), (51, 96), (23, 26), (96, 70)]$



(a) Proportion of empty and non_empty images in the dataset.



(b) Number of objects per non_empty images.

Obj per img	1	2	3	4	5	6	7	8	9	10	11	12	13	14	Total
Img w/ object	1479	148	48	14	8	4	3	0	1	0	0	1	0	1	1707
%	86,64	8,67	2,81	0,82	0,47	0,23	0,18	0,00	0,06	0,00	0,00	0,06	0,00	0,06	100
Img w/ fauna	269	46	13	10	5	1	1	0	1	0	0	1	0	1	348
%	77,30	13,22	3,74	2,87	1,44	0,29	0,29	0,00	0,29	0,00	0,00	0,29	0,00	0,29	100

(c) Distribution of images according to the number of objects.

Figure 3.3: Proportion and distribution of the images and objects in the SEMMACAPE dataset.

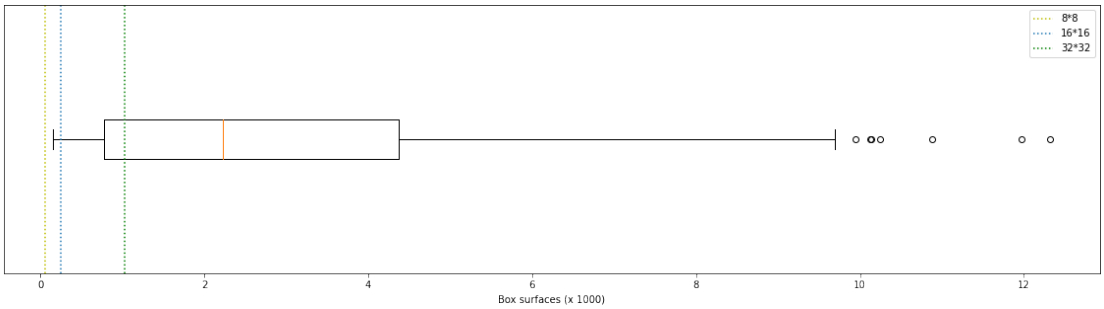


Figure 3.4: Bounding boxes surfaces.

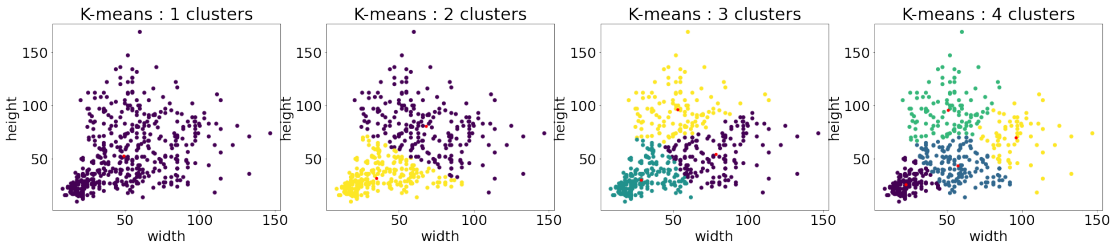


Figure 3.5: K-means clustering of the bounding boxes width and height. Red points: cluster centroids.

Chapter 4

Methods

4.1 Dataset

As indicated in Section 3, the objects belonging to the *fauna* class, numbering 513, are what this study aims to detect. They are distributed across 348 images out of 127 684. The proportion of 0.27% shows a major imbalance in the dataset, the consequences of which are explained in Section 2.4. To limit computing time while maintaining imbalanced data, the number of empty images used was reduced to 34 800, representing 27.6% of the initial pool.

Let F be the working dataset; it contains n_F = 35 148 images, corresponding to 27.8% of the total dataset. In F, the proportion of images containing at least one fauna object is 1%. Within this dataset, the images are then divided into 3 subsets, train T (n_T = 17 574), test I (n_I = 10 545), and validation V (n_V = 7 029), corresponding to 50%, 20% and 30% of F. In the interests of this research, as F is highly imbalanced, 50% of the non-empty images are in T, 30% in I, and 20% in V. We aim to avoid the case where T is made up entirely of empty images.

In the case of AL for OD, T is divided into 2 sub-parts, U and L, their definition being the same as in Section 2.1. The initial L will be distributed as follows, until otherwise indicated, L = 50 images with *fauna* objects; 50 empty images (n_L = 100). These images will be randomly selected from T. Finally, U is defined as: $U = T \backslash L$. All this is summarised in Table 4.1.

	Train T		Validation V	Test I	Total
	Labelled	Unlabelled			
F	17 574 (50%)		7 029 (20%)	10 545 (30%)	35 148 (100%)
	100	17 474			
non-empty images	174		69	105	348
	50	124			
empty images	17 400		6 960	10 440	34 800
	50	17 350			

Table 4.1: Distribution of objects in the dataset used for running experiments

4.2 Active Learning for Object Detection

The method implemented here for Object Detection by Active Learning is the one used in the article [6]. Section 2.3 already presents the mathematical concepts necessary to apply the method. This part presents the algorithm proposed in the article.

First, in the interest of efficiency and to leverage as much information as possible in each cycle, the images belonging to U that are to be added to L will be selected in batches. The initial U is randomly divided into q batches of size b. U can therefore be defined as follows: $U = U_1 \cup ... \cup U_i \cup ... \cup U_q$ with $i \in [1, q]$, $n_{U_i} = q$ and $\sum_{i=1}^{q} n_{U_i} = n_U$.

After this, the active learning cycle can be initialised. θ is trained with the initial L. Then predictions are made on all the images belonging to U. Each prediction is associated with a probability of belonging to the *fauna* class. The utility score is calculated for all these probabilities according to the sampling margin, defined by Equation 2.3. The D detection scores obtained per image are then aggregated using the methods detailed in Section 2.3, which could be the sum or average of the scores or maximum score for the image. A single score is obtained for each image.

It is now a question of selecting the batch of images which will be removed from U and added to L. The authors of [6] have decided to sum the scores of each image in the batch. This is also the solution that will be applied in future experiments. The winning batch will be the one with the highest sum of scores. This batch, U_{best}, will be considered the most informative. All the images from this batch are submitted for annotation. A human annotator or available ground truths are used to correct false positive and false negative predictions and validate the true positive and negative ones. The annotated pool of ten images U_{best} is added to L and removed from U. L is enhanced for the next training.

These few steps are repeated until a stop condition is encountered. This condition can be the image budget that can be annotated or the exhaustion of the images contained in U. This is described in the Algorithm 2, which is an adaptation of 1 to our specific problem.

Algorithm 2 Active Learning for Object Detection

Initialisation : Initial L, initial U, θ, budget b
$U = U_1, ..., U_q \leftarrow$ Split of U into q random batches of b
while $n_L < 250$ **do**
  train θ with L
  for all $x \in U$ **do**
    calculate $s(\theta, \mathbf{x})$ {using Sum, Average or Maximum}
  end for
  for all $U_i \in U$ **do**
    $s_{U_i} \leftarrow \sum_{x \in U_i} s(\theta, \mathbf{x})$
  end for
  $U_{best} \leftarrow \underset{i \in p}{\text{argmax}}\ U_i$
  annotate U_{best}
  $L \leftarrow L \cup (U_{best}, Y_{best})$
  $U \leftarrow U \backslash U_{best}$
end while

Chapter 5

Experiments and results

5.1 Experimental setup

Before running our experiments, we need to set the parameters and define what is compared. A baseline is set up as a reference for our comparisons. The experiments evaluate the performance of active learning for object detection with different types of utility score aggregation. Another variation to study is the impact of the distribution between empty and non-empty images within the initial L.

In both the baseline and the experiments related to active learning for object detection, the object detector used is Yolov7. The code used to train this object detector is the code provided by the authors of [35] and is available on Github[1]. The hyperparameters and the code relating to our dataset were tuned and adapted to have a model corresponding as closely as possible to our problem and the type of image analysed. This stage is described in more detail in Appendix A.1.

Baseline

The baseline consists of training the object detector, Yolov7, without active learning. Training is carried out with 250 images over 240 epochs, the optimizer is Stochastic Gradient Descent (SGD), and the learning rate scheduler is linear. The 250 images are randomly selected in T with a distribution of 125 images containing at least one *fauna* object and 125 empty images.

Experiments

Experiments carried out to evaluate the interest of Active Learning in the case of the detection of rare objects follow the methodology explained in Section 2.3. Initially, L is composed, as indicated above, by 50% of images containing at least one *fauna* object, denoted $L_{50\%}$. This pool is annotated beforehand by an expert.

The initial hypothesis is that it is possible for this expert, in terms of cost or time, to annotate only 250 images. This assumption also sets the condition for stopping the learning cycle. This number is arbitrary. However, it is justified by the fact that it is theoretically possible to add all the images containing at least one *fauna* object to L. The loop for the selection of the most informative images and the training of the model will therefore be repeated until n_L is equal to 250. The model will be set up in the same way as the baseline. The difference lies in the number

[1] `https://github.com/WongKinYiu/yolov7`

of epochs, 15, performed during each training session. To reach the stop condition, 16 cycles (240 epochs) are required.

The difference between each experiment lies in how the utility scores s_{MS} are aggregated within a single image. Each experiment uses a different aggregation method, such as sum, average, maximum and random selection, to select a batch of 10 images.

In a second phase, these four analyses will again be carried out with a different imbalanced distribution of the initial L. L comprises 10 images containing at least one object of interest and 90 empty images and notated $L_{10\%}$.

Evaluation The $\text{mAP}_{0.5}$, as described in Section 2.5, is given at each batch i.e. 10 new images. We evaluate the performances on the set I.

5.2 Results and discussion

5.2.1 Results

The performances are evaluated on test set I. This subset has never been used for any experimentation except for the evaluation of the model after adding each batch of images.

The results of the experiments in Figure 5.1 show that our baseline reaches an accuracy of 66% and is never exceeded by another method. Only the aggregation methods (sum, maximum and average) used on the $L_{50\%}$ tend to get close to this value. In this pool of experiments, after adding the 150 images, maximum aggregation methods outperform the random batch selection. The interest in selecting images according to informativeness is meaningful and improves the model's performance. Otherwise, among these methods, Figure 5.1a shows that none is better than the others. The experiments with the initial imbalanced labelled dataset perform poorly, as displayed in 5.1c. Without having more than 1% accuracy, all the methods have similar behaviour. None of the utility score aggregation methods takes a significant lead in performance. In comparison with the other pool of experiments, the random selection of images performs as well as using the sum of utility scores. Despite this poor performance, this shows that having an imbalance among the initial dataset doesn't bring enough information to the model. In this case, using active learning is useless.

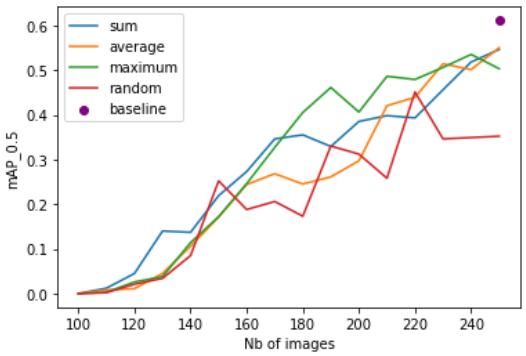
It is also interesting to look at the distribution of L when it reaches 250 images. This is displayed in Table 5.1 for the experimentation led on the equally distributed initial L. Of course, the baseline outperforms the AL proposed experiments. But the three utility score aggregation methods can reach a mAP0.5 of 55% with, on average, 57 images containing an object of interest against the 125 from the baseline training set.

Method	average	maximum	sum	random
Img w/ *fauna* objects	59	60	61	51
Empty img	191	190	189	199

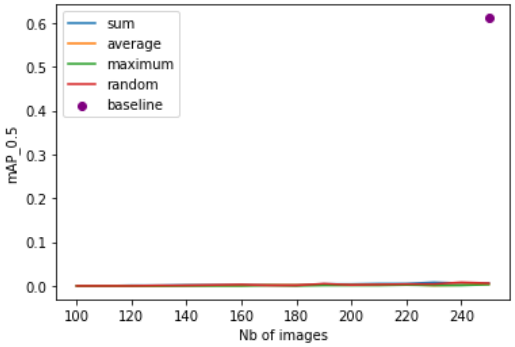
Table 5.1: Distribution of L after the active learning cycles.

5.2.2 Discussion

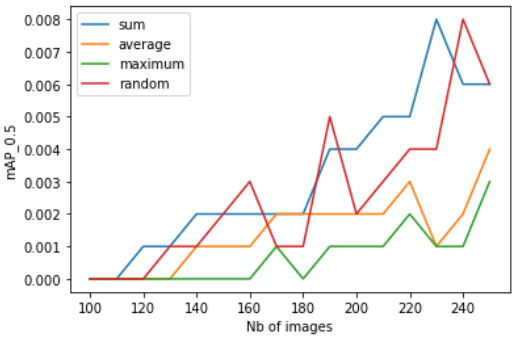
The method used here to test the interest in adding active learning in object detection tasks can be improved and made more suitable for imbalanced data. The CE loss function was used in these experiments. The use of FL, a loss function better adapted to imbalanced problems,



(a) Comparison between baseline and aggregation methods using active learning on YOLOv7 object detector with initial labelled set $L_{50\%}$



(b) Comparison between baseline and aggregation methods using active learning on YOLOv7 object detector with initial labelled set $L_{10\%}$



(c) Comparison of aggregation methods using active learning on YOLOv7 object detector with initial labelled set $L_{10\%}$

		Initial labelled set $L_{10\%}$				Initial labelled set $L_{10\%}$			
Experiment	baseline	average	maximum	sum	random	average	maximum	sum	random
$mAP_{0.5}$	**61.1%**	55.0%	50.3%	54.6%	35.2%	0.4%	0.3%	0.6%	0.6%

(d) Evaluation of the performance on the test dataset I for the different methods experimented

Figure 5.1: Comparison of the proposed methods

could be studied. It is also possible to work on optimising the initial L. Intuitively, we could ask ourselves what would happen if the images in L were randomly selected. According to the composition of our dataset and the results obtained previously, the number of images containing objects would be in the minority. Thus, poor performance would be expected. One idea for optimising L would be to determine from the beginning which images are the more informative to initialise our learning with as much information as possible.

Adjusting the size of batches also seems to be a way of improving our model. In our previous experiments, an average of 7 images with objects were added to L, which is few. This number could be increased by reducing the size of batches of images. By always having the same budget b of images to label at each stage, U could be divided into batches of size b/t. After calculating the score of all the batches, the t batches of images providing the most information will be added to L.

Chapter 6

Conclusion

In this report, we tried to assess the integration of active learning for object detection tasks on a highly imbalanced dataset. We evaluate the utility score aggregation methods on the SEMMACAPE dataset. Our case scenario is the detection of marine megafauna objects in the ocean where the background is predominant. We show that using uncertainty sampling in a pool-based active learning method is efficient even if it did not lead to an outperformance of the baseline.

Using active learning in case of imbalanced data is meaningful. Indeed, if the imbalance in the dataset has consequences on the performance of the model, this latter can still perform well. Nevertheless, the imbalance that must be avoided is the one inside the initially labelled dataset.

Annexes

A.1 Hyperparameters of YOLOv7

The object detector, Yolov7, needs to be customised to fit our dataset as closely as possible and to be as efficient as possible. The implementation provided by the authors of [35] is adapted to the MicroSoft Common Objects in COntext (MSCOCO) dataset[1]. A simple example of the difference between our two dataset is the size of the images, 640 pixels for MSCOCO and 416 for the SEMMACAPE dataset. These model's hyperparameters need to be modified and tuned.

To obtain an optimised value for each hyperparameter, the *evolve* function is used. This is a function that runs through several values for the same parameter and determines the optimum value. To obtain a representative value, the analysis is carried out on 10 different distributions (train/test/val) of all the images containing at least one *fauna* object. The mean and standard deviation are calculated for each of the model's 31 hyperparameters. The values chosen to configure the model are displayed in Table 1.

[1] cocodataset.org

distribution	1	2	3	4	5	6	7	8	9	10	ecart-type	average	evolved
lr0	0,00977	0,00761	0,0102	0,0109	0,00744	0,0101	0,0106	0,0118	0,0103	0,0152	0,002058445	0,010392	0,01
lrf	0,0919	0,0897	0,135	0,11	0,0989	0,0657	0,084	0,115	0,106	0,0707	0,019850262	0,09669	0,097
momentum	0,873	0,98	0,945	0,861	0,882	0,871	0,905	0,875	0,958	0,961	0,042828612	0,9111	0,911
weight_decay	0,00055	0,00047	0,00044	0,00056	0,00056	0,00057	0,0005	0,0005	0,00052	0,00028	8,22E-05	0,000495	0,0005
warmup_epochs	2,73	2,64	3,36	2,81	3,68	4,4	2,96	2,2	2,56	2,37	0,633584249	2,971	3
warmup_momentum	0,93	0,871	0,758	0,922	0,95	0,543	0,603	0,772	0,651	0,805	0,136071489	0,7805	0,78
warmup_bias_lr	0,0958	0,0797	0,109	0,146	0,0989	0,104	0,108	0,134	0,0963	0,0978	0,018437584	0,10695	0,1
box	0,0444	0,0363	0,0583	0,0358	0,04	0,0498	0,0483	0,0433	0,0391	0,0604	0,008173867	0,04557	0,046
cls	0,268	0,292	0,408	0,381	0,306	0,27	0,278	0,495	0,229	0,481	0,089394407	0,3408	0,34
cls_pw	1,1	0,927	0,976	0,696	0,586	1,12	1	1,52	1,05	1,93	0,367425435	1,0905	1,09
obj	0,605	0,447	0,81	0,72	1,1	1,08	0,738	0,674	0,767	0,527	0,201139156	0,7468	0,75
obj_pw	0,961	0,678	1,12	1,18	0,77	0,81	1,05	1,04	0,952	1,32	0,18665286	0,9881	0,98
iou_t	0,2	0,2	0,2	0,2	0,2	0,2	0,2	0,2	0,2	0,2	2,78E-17	0,2	0,2
anchor_t	3,48	3,86	3,38	5,15	4,3	3,09	4,41	5,78	4,28	3,4	0,810074688	4,113	4
fl_gamma	0	0	0	0	0	0	0	0	0	0	0	0	0
hsv_h	0,015	0,015	0,015	0,015	0,015	0,015	0,015	0,015	0,015	0,015	3,47E-18	0,015	0,015
hsv_s	0,542	0,683	0,626	0,773	0,849	0,595	0,633	0,9	0,502	0,545	0,128301052	0,6648	0,7
hsv_v	0,446	0,412	0,351	0,262	0,563	0,222	0,4	0,344	0,408	0,327	0,091358908	0,3735	0,4
degrees	0	0	0	0	0	0	0	0	0	0	0	0	0
translate	0,218	0,251	0,197	0,181	0,26	0,205	0,221	0,102	0,257	0,216	0,043935862	0,2108	0,2
scale	0,641	0,74	0,565	0,519	0,287	0,393	0,5	0,582	0,397	0,556	0,124945588	0,518	0,5
shear	0	0	0	0	0	0	0	0	0	0	0	0	0
perspective	0	0	0	0	0	0	0	0	0	0	0	0	0
flipud	0	0	0	0	0	0	0	0	0	0	0	0	0
fliplr	0,52	0,542	0,553	0,368	0,453	0,293	0,487	0,501	0,557	0,366	0,087103387	0,464	0,5
mosaic	1	1	1	1	1	1	1	1	1	1	0	1	1
mixup	0	0	0	0	0	0	0	0	0	0	0	0	0
copy_paste	0	0	0	0	0	0	0	0	0	0	0	0	0
paste_in	0	0	0	0	0	0	0	0	0	0	0	0	0
loss_ota	0,971	0,873	0,874	0,498	0,397	0,766	1	0,815	0,966	0,767	0,190565501	0,7927	0,79
anchors	3	3	3	3	3	3	3	3	3	3	0	3	3

Table 1: Hyperparameters tuning

Glossary

AI Artificial Intelligence.

AL Active Learning.

AUC Area Under the Curve.

CE Cross Entropy.

CNN Convolutional Neural Networks.

FL Focal Loss.

IoU Intersection-over-Union.

IRISA Institute for Research in Computer Science and Random Systems.

mAP mean Average Precision.

ML Machine Learning.

MSCOCO MicroSoft Common Objects in COntext.

OD Object Detection.

OFB French Office for Biodiversity.

PR Precision-Recall.

R-CNN Region based Convolutional Neural Networks.

SEMMACAPE (Suivi et Étude de la Mégafaune Marine par Caractérisation Automatique dans les Parcs Éoliens).

UBS University of Southern Brittany.

YOLO You Only Look Once.

Bibliography

[1] Dana Angluin. Queries and concept learning. *Machine Learning*, 2(4):319–342, 1988.

[2] Mohamed Bekkar and Taklit Akrouf Alitouche. Imbalanced Data Learning Approaches Review. *International Journal of Data Mining & Knowledge Management Process*, 3(4):15–33, 2013.

[3] Paul Berg, Deise Santana Maia, Minh-Tan Pham, and Sébastien Lefèvre. Weakly Supervised Detection of Marine Animals in High Resolution Aerial Images. *Remote Sensing*, 14(2):339, 2022.

[4] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. YOLOv4: Optimal Speed and Accuracy of Object Detection, 2020.

[5] Justine Boulent, Bertrand Charry, Malcolm McHugh Kennedy, Emily Tissier, Raina Fan, Marianne Marcoux, Cortney A. Watt, and Antoine Gagné-Turcotte. Scaling whale monitoring using deep learning: A human-in-the-loop solution for analyzing aerial datasets. *Frontiers in Marine Science*, 10:1099479, 2023.

[6] Clemens-Alexander Brust, Christoph Käding, and Joachim Denzler. Active Learning for Deep Object Detection:. In *Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, pages 181–190, Prague, Czech Republic, 2019. SCITEPRESS - Science and Technology Publications.

[7] Jiwoong Choi, Ismail Elezi, Hyuk-Jae Lee, Clement Farabet, and Jose M Alvarez. Active learning for deep object detection via probabilistic modeling, 2021.

[8] David Cohn, Les Atlas, and Richard Ladner. Improving generalization with active learning. *Machine Learning*, 15(2):201–221, 1994.

[9] Tausif Diwan, G. Anirudh, and Jitendra V. Tembhurne. Object detection using YOLO: challenges, architectural successors, datasets and applications. *Multimedia Tools and Applications*, 82(6):9243–9275, 2023.

[10] Zhanpeng Feng, Shiliang Zhang, Rinyoichi Takezoe, Wenze Hu, Manmohan Chandraker, Li-Jia Li, Vijay K Narayanan, and Xiaoyu Wang. Albench: A framework for evaluating active learning in object detection, 2022.

[11] Chloé Friguet, Romain Dambreville, Ewa Kijak, Mathieu Laroze, and Sébastien Lefèvre. Data annotation with active learning: application to environmental surveys. 2020.

[12] JuiHsi Fu and SingLing Lee. Certainty-based active learning for sampling imbalanced datasets. *Neurocomputing*, 119:350–358, 2013.

[13] Yifan Fu, Xingquan Zhu, and Bin Li. A survey on instance selection for active learning. *Knowledge and Information Systems*, 35(2):249–283, 2013.

[14] Haibo He and E.A. Garcia. Learning from Imbalanced Data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1263–1284, 2009.

[15] Guo Haixiang, Li Yijing, Jennifer Shang, Gu Mingyun, Huang Yuanyue, and Gong Bing. Learning from class-imbalanced data: Review of methods and applications. *Expert Systems with Applications*, 73:220–239, 2017.

[16] Elmar Haussmann, Michele Fenzi, Kashyap Chitta, Jan Ivanecky, Hanson Xu, Donna Roy, Akshita Mittel, Nicolas Koumchatzky, Clement Farabet, and Jose M Alvarez. Scalable active learning for object detection, 2020.

[17] Vishal Kaushal, Rishabh Iyer, Suraj Kothawade, Rohan Mahadev, Khoshrav Doctor, and Ganesh Ramakrishnan. Learning from less data: A unified data subset selection and active learning framework for computer vision, 2019.

[18] Suraj Kothawade, Saikat Ghosh, Sumit Shekhar, Yu Xiang, and Rishabh Iyer. Talisman: Targeted Active Learning for Object Detection with Rare Classes and Slices Using Submodular Mutual Information. In Shai Avidan, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner, editors, *Computer Vision – ECCV 2022*, volume 13698, pages 1–16. Springer Nature Switzerland, Cham, 2022.

[19] Mathieu Laroze, Romain Dambreville, Chloe Friguet, Ewa Kijak, and Sebastien Lefevre. Active Learning to Assist Annotation of Aerial Images in Environmental Surveys. In *2018 International Conference on Content-Based Multimedia Indexing (CBMI)*, pages 1–6, La Rochelle, 2018. IEEE.

[20] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection, 2017.

[21] Hossin M and Sulaiman M.N. A Review on Evaluation Metrics for Data Classification Evaluations. *International Journal of Data Mining & Knowledge Management Process*, 5(2):01–11, 2015.

[22] N. Memarsadeghi and D.P. O'Leary. Classified information: The data clustering problem. *Computing in Science & Engineering*, 5(5):54–60, 2003.

[23] Vu-Linh Nguyen, Mohammad Hossein Shaker, and Eyke Hüllermeier. How to measure uncertainty in uncertainty sampling for active learning. *Machine Learning*, 111(1):89–122, 2022.

[24] Kemal Oksuz, Baris Can Cam, Sinan Kalkan, and Emre Akbas. Imbalance Problems in Object Detection: A Review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(10):3388–3415, 2021.

[25] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun, 2015.

[26] Jan Schneegans, Maarten Bieshaar, and Bernhard Sick. A practical evaluation of active learning approaches for object detection. 2022.

[27] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach, 2017.

[28] Burr Settles. Active Learning Literature Survey. 2010.

[29] Wissam Siblini, Jordan Fréry, Liyun He-Guelton, Frédéric Oblé, and Yi-Qing Wang. Master your Metrics with Calibration. In Michael R. Berthold, Ad Feelders, and Georg Krempl, editors, *Advances in Intelligent Data Analysis XVIII*, volume 12080, pages 457–469. Springer International Publishing, Cham, 2020.

[30] Mingxing Tan, Ruoming Pang, and Quoc V Le. Efficientdet: Scalable and efficient object detection, 2020.

[31] Holger Trittenbach, Adrian Englhardt, and Klemens Böhm. An overview and a benchmark of active learning for outlier detection with one-class classifiers. *Expert Systems with Applications*, 168:114372, 2021.

[32] Devis Tuia, Michele Volpi, Loris Copa, Mikhail Kanevski, and Jordi Munoz-Mari. A Survey of Active Learning Algorithms for Supervised Remote Sensing Image Classification. *IEEE Journal of Selected Topics in Signal Processing*, 5(3):606–617, 2011.

[33] Adam Van Etten. Satellite imagery multiscale rapid detection with windowed networks. In *2019 IEEE winter conference on applications of computer vision (WACV)*, pages 735–743. IEEE, 2019.

[34] Huy V. Vo, Oriane Siméoni, Spyros Gidaris, Andrei Bursuc, Patrick Pérez, and Jean Ponce. Active Learning Strategies for Weakly-Supervised Object Detection. In Shai Avidan, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner, editors, *Computer Vision – ECCV 2022*, volume 13690, pages 211–230. Springer Nature Switzerland, Cham, 2022.

[35] Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors, 2023.

[36] Hualong Yu, Xibei Yang, Shang Zheng, and Changyin Sun. Active Learning From Imbalanced Data: A Solution of Online Weighted Extreme Learning Machine. *IEEE Transactions on Neural Networks and Learning Systems*, 30(4):1088–1103, 2019.

[37] Wenan Yuan. Accuracy Comparison of YOLOv7 and YOLOv4 Regarding Image Annotation Quality for Apple Flower Bud Classification. *AgriEngineering*, 5(1):413–424, 2023.

[38] Yisong Yue, Thomas Finley, Filip Radlinski, and Thorsten Joachims. A support vector method for optimizing average precision. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 271–278, Amsterdam The Netherlands, 2007. ACM.

[39] Zhong-Qiu Zhao, Peng Zheng, Shou-Tao Xu, and Xindong Wu. Object Detection With Deep Learning: A Review. *IEEE Transactions on Neural Networks and Learning Systems*, 30(11):3212–3232, 2019.

[40] Wenbo Zhu, Quan Wang, Lufeng Luo, Yunzhi Zhang, Qinghua Lu, Wei-Chang Yeh, and Jiancheng Liang. Cpam: Cross patch attention module for complex texture tile block defect detection. *Applied Sciences*, 12:11959, 11 2022.