



MASTER 1 INFORMATIQUE - TER

Système de reconnaissance
d'individus pour une espèce de
baleines noires en danger
d'extinction

*Hugo Maitre
Thomas Lecampion
Adrien Linares*

Encadrant :
Pr. William PUECH

1^{er} mai 2022

Table des matières

1	Préambule	2
1.1	Contexte général	2
1.1.1	Données	3
1.1.2	Résultat attendu par Kaggle	7
1.2	Organisation du travail	7
1.2.1	Gestion de projet	7
1.2.2	Plannification des tâches et planning	7
1.3	Motivations	8
1.4	Difficultés majeures	9
2	Réalisation du projet	11
2.1	Introduction sur la stratégie effectuée	11
2.2	Choix techniques	11
2.2.1	Python et ses librairies	11
2.2.2	Stockage des images	11
2.2.3	Boostage des calculs GPU	12
2.3	Développement des modèles de deep learning	12
2.3.1	Object Localization Yolo v3	12
2.3.1.1	Premiers prétraitements	15
2.3.1.2	Développement du premier modèle	17
2.3.1.3	Évaluation des résultats	18
2.3.1.4	Problèmes et seconds prétraitements	19
2.3.2	Head Aligner Yolo v4 et Algo d'Alignement	20
2.3.2.1	Premiers prétraitements	20
2.3.2.2	Développement du second modèle	20
2.3.2.3	Résultats	20
2.3.2.4	Alignment des images	20
2.3.3	Système de reconnaissance faciale	21
3	Conclusion	22
3.1	Perspectives d'amélioration	22
3.2	Ce qu'on a appris	22
	Bibliographie	23

1 Préambule

1.1 Contexte général

Les baleines noires de l'Atlantique Nord sont une espèce de cétacés qui se distinguent des autres espèces de baleines entre autres par des callosités blanches se trouvant sur leurs têtes, qui est en fait de la peau kératinisée. Cette espèce de baleine, comme son nom l'indique, vit dans l'Océan Atlantique et à plus forte raison entre les degrés 20 et 60 de latitude Nord avec des déplacements variant selon les saisons.

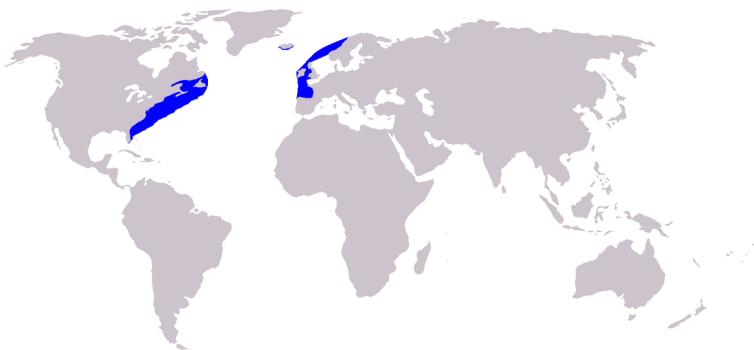


FIGURE 1 – Répartition géographique de l'espèce

Elles sont connues pour leurs comportements sociable vis à vis de l'humain. Elle se déplace lentement et souvent à la surface de l'eau. On estime qu'entre 1991 et 2007, 50% des mortalités seraient dues à des collisions avec les bateaux[1]. Elle est également chassée principalement pour son huile, ce qui fait d'elle l'une des grandes baleines les plus menacées au monde. Aujourd'hui, sa population est estimée à moins de 450 individus, elle est classifiée par l'IUCN¹ comme en danger critique d'extinction.

1. Union internationale pour la conservation de la nature



FIGURE 2 – Statut de conservation établi par l’IUCN

C'est dans ce contexte que la NOAA (National Oceanic and Atmospheric Administration) a proposé un challenge d'une valeur de \$10.000 en 2016, afin de construire un système de reconnaissance automatique de ces baleines à partir de méthodes de machine learning. En effet, seulement quelques personnes dans le monde savent aujourd’hui distinguer ces baleines à l’œil nu, la majorité étant des chercheurs expérimentés dans le domaine. C'est de plus un processus qui prend beaucoup de temps. Cet outil sera donc nécessaire pour surveiller bien plus aisément la population de baleines noires de l’Atlantique Nord et de même avoir accès à son historique de santé. Le challenge a été publié sur Kaggle, site web rassemblant une communauté de data scientists et professionnels de l’apprentissage automatique, leur permettant de participer à des concours sur des problèmes réels, à partir de données réelles. L’abréviation AN sera utilisée dans la suite de ce rapport pour « Atlantique Nord ».

1.1.1 Données

Nous avons à notre disposition 11468 images aériennes de baleines noires de l’A.N. La qualité des images est très aléatoire, on constate par exemple des images 200*200 pixels côtoyant des images de très haute définition, images de 4000*3000 pixels. L’exposition des images est aussi très aléatoire même si l’on constate que 90% des images sont correctement exposées. Certaines images souffrent de contraste trop fort ou inversement apparaissent trop “lisses”. On peut voir ci-dessous ces

écart sur un échantillon non aléatoire du dataset.

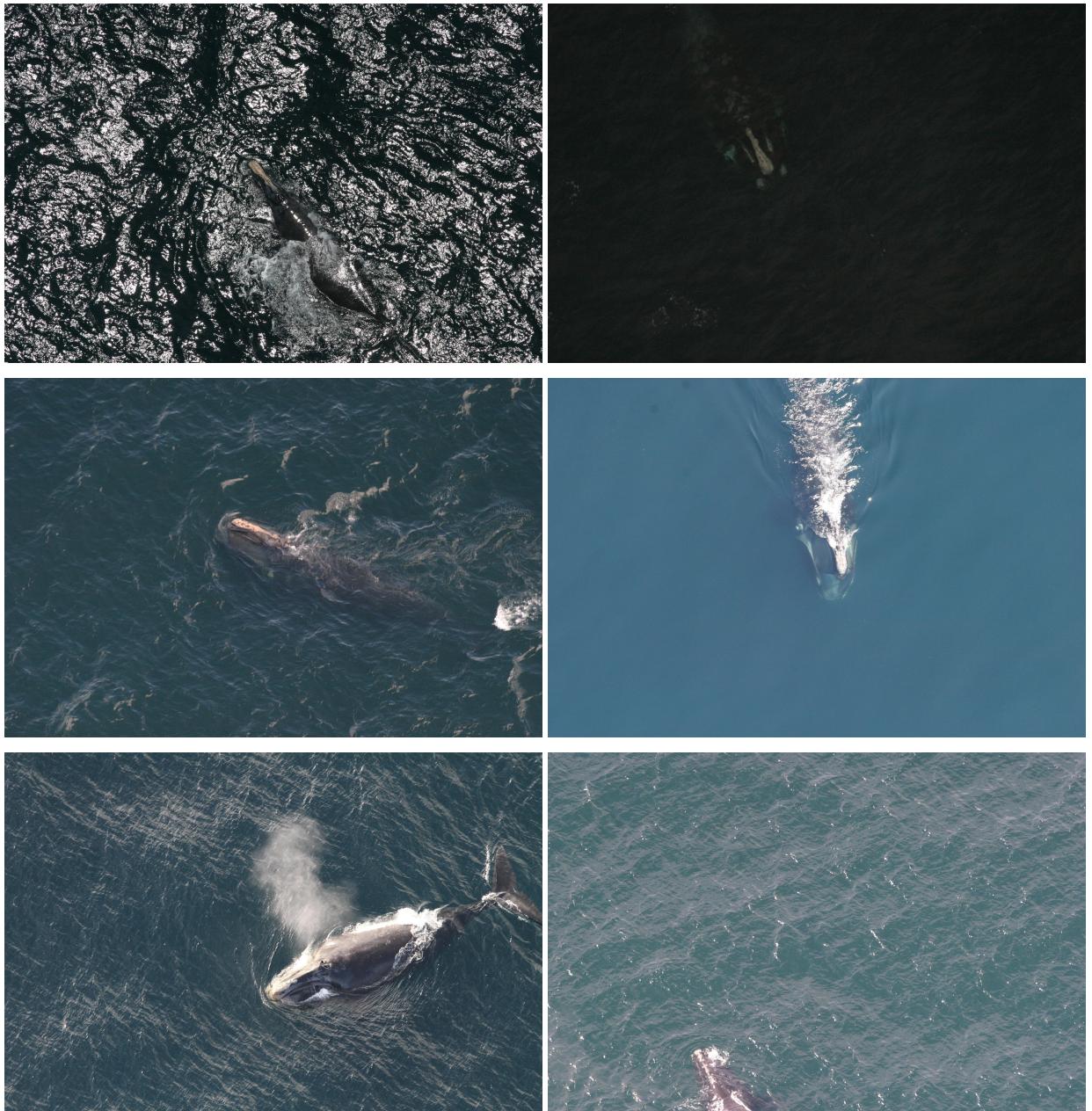


FIGURE 3 – Echantillon non aléatoire du Dataset

Parmi le dataset nous avons seulement 4544 images qui sont labellisées, c'est-à-dire environ 40% du dataset. Nous n'avons bien évidemment pas accès aux labels des images restantes qui serviront à évaluer les performances de notre classifieur

lors de la soumission de nos prédictions en ligne sur kaggle. On comprend déjà la difficulté qui nous attend : le manque de données. On compte 447 individus de baleines noires répertoriées ce qui est probablement très proche du nombre de baleines noires de l'A.N existantes. Parmi les images labellisées on constate un nombre très hétérogène d'images attribuées à nos baleines. En effet certaines d'entre elles sont des "célébrités" ayant une cinquantaine d'images rien que pour elle mais la majorité des individus ont seulement entre 1 et 10 images.

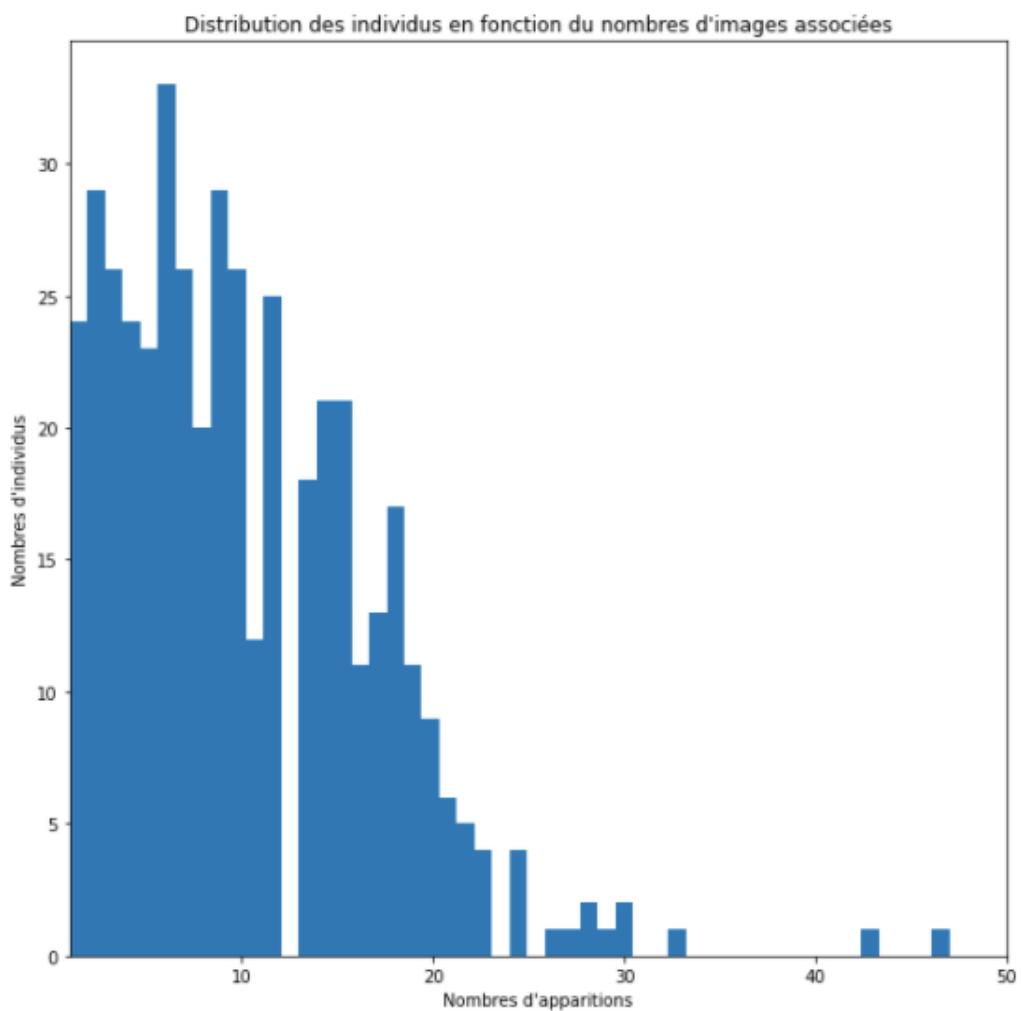


FIGURE 4 – Visualisation de l'hétérogéité des occurrences

1.1.2 Résultat attendu par Kaggle

La solution s'imposant assez naturellement pour ce type de problème est d'utiliser une série de réseaux de neurones convolutifs. La plateforme met à notre disposition une "vérité terrain" c'est-à-dire l'identité de 4544 images et nous évaluera sur les prédictions effectuées sur les 6924 images restantes. La plateforme de Kaggle nous évaluera à la fin sur la précision, f1-score et rappel de notre modèle. À titre d'exemple, le modèle ayant le mieux performé est d'environ 80%[2].

1.2 Organisation du travail

1.2.1 Gestion de projet

Nous avons eu l'idée de nous servir de Miro, outil collaboratif de travail mais qui a été au final très peu utilisé. En ce qui concerne la méthodologie de gestion de projet, nous avons choisi la méthode agile qui nous a semblé la plus adaptée ici. Cette dernière a un usage plus particulier dans le contexte du deep learning et celui plus général du machine learning La communication et les réunions se font sur Discord ou en présentiel. Les rôles au sein de notre équipe ont été définis comme suit :

- **Hugo Maître** : chef de projet et développeur
- **Adrien Linares** : développeur
- **Thomas Lecampion** : développeur

1.2.2 Plannification des tâches et planning

Afin de mener à bien ce projet, nous avons utilisé différents outils nécessaires à l'organisation des tâches et de l'avancement dans le temps du projet. Nous avons premièrement utilisé Gantt, qui était avant tout un planning prévisionnel, dont

voici ci-dessous une capture :

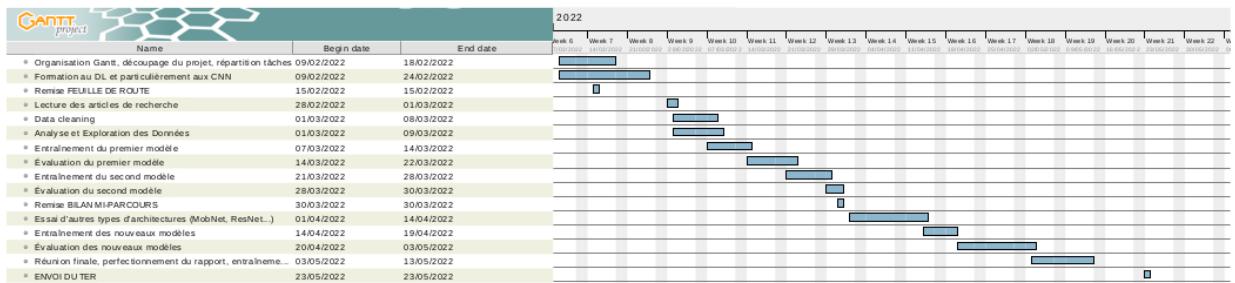


FIGURE 5 – Planning prévisionnel

1.3 Motivations

Nous avons recueilli les témoignages de chacun d'entre nous afin de savoir pourquoi nous avons, à l'unanimité, choisi un tel projet et pas un autre.

- **Thomas Lecampion** : J'ai choisi de rejoindre mes collègues sur ce TER pour m'initier au machine learning. En effet, c'est un domaine dans lequel je n'ai littéralement aucune expérience avant l'affectation des sujets. Je souhaitais apprendre pour réaliser certains projets personnels.
- **Adrien Linares** : Ce projet m'a semblé le compromis idéal sur des choses que j'affectionne particulièrement, d'une part, la protection de la biodiversité, de l'autre, le domaine du deep learning et du traitement d'images. Il me servira en effet de tremplin pour de futurs stages et emploi en m'apprenant davantage sur ces domaines.
- **Hugo Maître** : À remplir

En complément de ce qui a été dit, nous avions également hésité avec un autre

projet portant sur du NLP mais le voyant déjà en cours nous avons choisi de faire quelque chose de nouveau, portant donc sur le Computer vision cette fois-ci.

1.4 Difficultés majeures

Dès le commencement, nous avons pu distinguer sur le projet trois difficultés majeures :

Premièrement, il est nécessaire d'uniquement récupérer les informations pertinentes sur chacune des images de notre dataset, isoler les baleines. Autrement dit, il nous faut un moyen de localiser et récupérer la zone autour d'une baleine présente sur une image. La difficulté à première vue réside dans le fait que les baleines sont prises sous différents angles de vues plus ou moins loin et n'occupe donc pas du tout le même espace sur l'image. Pour cela nous envisageons d'entraîner un réseau de neurones (certainement un des modèles YOLO) capable de détecter automatiquement des objets sur une image. Une fois la baleine détectée et isolée il faudra certainement établir un deuxième réseau de neurones capable de reconnaître les différentes caractéristiques de la baleine probablement la tête et l'évent. Là aussi en regardant un petit peu nos images on s'aperçoit que l'écume blanche sur les images va grandement complexifier notre tâche puisque l'évent² ou le bout de la tête sont des zones qui apparaissent aussi blanches comme l'écume aussi bien qu'il est parfois difficile même pour un humain de les repérer correctement sur l'image. De plus, les baleines peuvent se présenter sous différents profils. Ce qui rend leur identification difficile même pour un spécialiste. *images de baleines : (image où le dos est nette, gueule ouverte, etc...)*

Ensuite, nous devrons effectuer quelques traitements sur les images réduites ob-

2. narine double des cétacés

tenues afin d'en faire de véritables photos d'identités, standardisées à la manière d'image d'identité humaine ou les caractéristiques des baleines devront matcher une certaine zone de l'image. Il s'agira de corriger l'orientation des têtes. La difficulté n'est pas tant l'opération en elle-même mais plutôt de savoir de quel angle l'image doit-elle être orientée. Pointer l'emplacement exact du bout de la tête et de l'évent nous donne l'angle correspondant par de simples calculs. Enfin, il y a de fortes différences d'intensité lumineuse, de contrastes d'une image à l'autre. Un rehaussement de contraste par égalisation d'histogramme ainsi qu'une corrections de l'exposition des images doit sans doute considérablement réduire les écarts.

Enfin, il faut implementer un dernier réseau bien différent des 2 autres pour la tâche de reconnaissance faciale. Ce réseau doit être capable de détecter quel est l'individu donné en entrée. Le principal problème de cette partie provient du grand déséquilibre de la quantité de nos données comme expliqué on peut le voir sur le diagramme (partie 1.1.1). De plus, les baleines apparaissent souvent dans des conditions bien différentes. Certaines ouvrent la gueule, sont entièrement sous l'eau ou ne possèdent qu'une photo en basse résolution, trop sombre avec un gros problème de mise au point...

2 Réalisation du projet

À remplir

2.1 Introduction sur la stratégie effectuée

2.2 Choix techniques

2.2.1 Python et ses librairies

Parmi les différentes options qui s'offrait à nous le langage de programmation Python semblait de loin être le plus adapté pour les tâches de computer Vision. Le langage dispose d'une immense communauté surtout dans le domaine du deep learning appliqué à l'image, il propose des solutions rapides et faciles à mettre en place.

Parmi les librairies souvent utilisées j'en citerai quelques-unes. Opencv : pour toutes les tâches de manipulations d'images, crop rotation ainsi que des opérations plus complexes (homographie, corrections d'histogramme).

Numpy : Librairie permettant la manipulation plus rapide et performante de tableaux (et donc d'images) par le processus de "vectorization".

Matplotlib : une des plus grandes librairie de la communauté très utilisée pour la visualisation de données, pour tracer des graphiques etc...

Enfin tensorflow utilisé pour la mise en place de réseau de neurones.

2.2.2 Stockage des images

L'ensemble des données représente environ 10 GB d'images ce qui n'est pas si conséquent que ça mais le problème étant qu'aucun de nous avait un ordinateur assez performant pour entraîner les modèles. Le passage par un outil tiers s'est donc imposé. J'ai donc dû charger sur un Google Drive plusieurs GB d'images en

ligne pour permettre un entraînement via des GPU plus puissants.

2.2.3 Boostage des calculs GPU

Les algorithmes de deep learning utilisé ont tous demandé à la fois beaucoup de puissance de calcul GPU mais aussi beaucoup de RAM GPU. J'ai donc souscrit à un abonnement Google Colab Pro qui met à disposition ce type de ressources (sans être trop limité). Tous les entraînements des modèles se sont donc effectués sur Google Colab.

2.3 Développement des modèles de deep learning

2.3.1 Object Localization Yolo v3

Le premier réseau implémenté va servir à détecter la baleine sur l'image comme les baleines sont photographiées de plus ou moins loin cette étape est crucial pour isoler seulement l'information qui nous est nécessaire sur l'image.

Nous nous intéressons dans un premier temps à l'algorithme Yolo "You Only Look Once". Cet algorithme est notamment très utilisé pour des applications de détection d'objets sur des images ou vidéo. Sa réputation est en partie due à sa rapidité et sa précision. Il est aujourd'hui copieusement utilisé dans de nombreux domaines comme celui des voitures autonomes ou du tracking d'objet sur vidéo.

Les créateurs mettent à disposition de nombreux datasets pour entraîner un modèle sur des classes assez courantes (voitures, humain,...) mais comme vous vous en doutez certainement le réseau n'a pas été entraîné sur des baleines noires de l'A.N. Nous allons donc utiliser ce réseau qui est préentraîné sur des datasets classiques ImageNet pour le premier algorithme présenté dans le papier de recherche, puis l'entraîner avec nos propres objets.

L'algorithme fonctionne à l'aide d' "anchor boxes" c'est à dire de petites boites rectangulaire qui vont aider à la détection de features qui vont aider à la détection de l'objet. Elles sont définies comme suit :

$$y = \begin{pmatrix} p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ . \\ . \\ c_n \end{pmatrix}$$

$p_c = 1$ si il y'a un objet 0 sinon

b_x, b_y les coordonnées du centre du rectangle

b_w, b_h la largeur et hauteur

$n = \#\text{classes}$

Par exemple Yolov3 utilise 9 formes possibles pour les anchor boxes. Il s'avère que la plupart de ces boîtes de délimitation ont certains rapports entre leur hauteur et leur largeur. Donc, au lieu de prédire directement une boîte englobante (bounding box), YOLOv3 prédit des ensembles de boîtes avec des rapports hauteur sur largeur particuliers, ces ensembles de boîtes pré-déterminés sont les "anchor box". YOLOv3 dispose de trois couches finales, la première a une dimension divisée par 31 par rapport à l'image initiale, la deuxième par 16 et la troisième par 8. Ainsi en partant d'une image de taille 416×416 pixels, les trois features maps en sortie du réseau auront des tailles respectives de 13×13 , 26×26 et 52×52 pixels. C'est en

ce sens que YOLOv3 prédit trois niveaux de détails, pour détecter respectivement les gros, moyens et petits objets.

Partant d'une image de taille 416×416 pixels, chaque entrée (pixel) est « conduit » à travers le réseau jusqu'à trois cellules (première sortie, 2ème, 3ème). Pour chaque cellule trois bounding box sont prédites, cela en fait un total de 9 qui sont issues des 9 "anchor box". Pour chaque bounding box, un score d'objectness et des scores d'appartenance aux classes sont prédits. Au total, le réseau propose $52 \times 52 \times 3 + 26 \times 26 \times 3 + 13 \times 13 \times 3 = 10647$ bounding boxes.

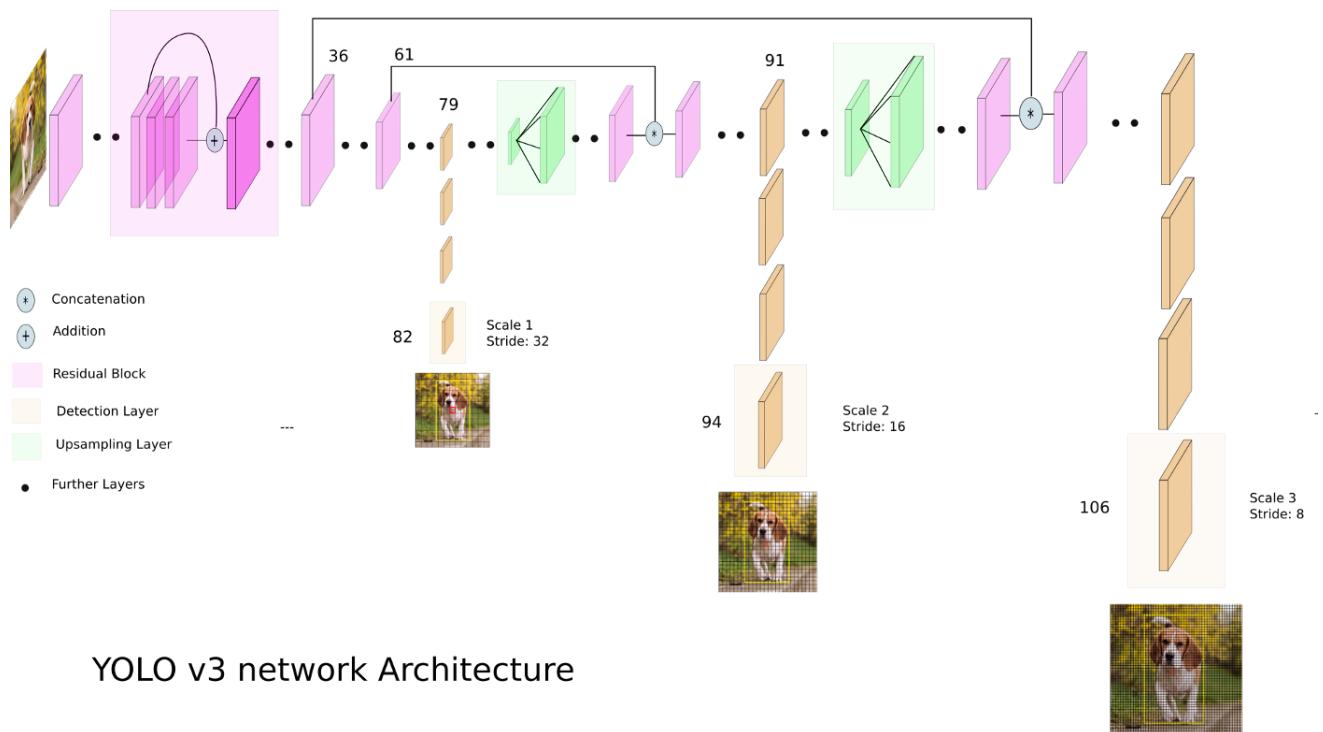


FIGURE 6 – Architecture Yolov3

image libre de droit src : <https://towardsdatascience.com/yolo-v3-object-detection-53fb7d3bfe6b>
L'IoU – ou Intersection over Union – est un indicateur de la qualité de la détection

d'un objet en comparant, dans un dataset d'entraînement, la position connue de l'objet dans l'image avec la prédiction faite par l'algorithme. L'IoU est le rapport entre la surface de l'intersection de 2 bounding box (celle prédite et la véritable) considérées et la surface de l'union des 2.

L'IoU peut valoir entre 0 (pour une détection totalement ratée) et 1 (pour une détection parfaite). De façon générale l'objet est considéré comme étant détecté à partir d'un IoU supérieur à un certain nombre (0.24 dans la littérature).

Nous utiliserons abondamment une métrique très similaire à l'IOU comme mesure pour notre validation lors de l'entraînement. La métrique mAP basé sur la moyenne des précisions. Pour cette première tâche nous optons pour la version3 de l'algorithme.

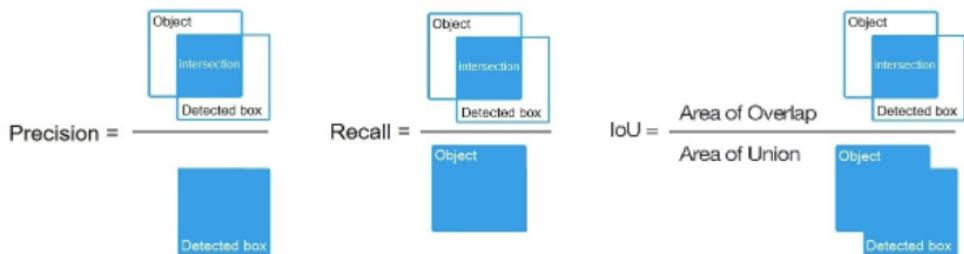


FIGURE 7 – Precision - Recall - IOU

image libre de droit src : <https://github.com/AlexeyAB/darknet>

Plus de détail sur le papier de recherche de YOLO³.

2.3.1.1 Premiers prétraitements Nous avons choisi LabelImg pour annoter nos données. L'étape de l'annotation est une étape cruciale et à un impact immense sur la qualité des résultats de prédiction. Cette étape consiste à tracer des

3. <https://arxiv.org/pdf/1506.02640.pdf>

rectangles autour de l'objet que nous souhaitons détecter et d'annoter un nom de classe à chaque rectangle sur l'image. Pour constituer notre jeu d'entraînement nous avons donc choisi dans un premier temps de tracer les box autour de la baleine entière sur environ 500 images. Nous verrons plus tard que c'était d'une mauvaise idée.

Le logiciel génère pour chaque image un fichier txt correspondant où chaque ligne correspond à un rectangle avec comme paramètre cet respectivement cet ordre : $\text{classe}_i, d, x, y, h, w$

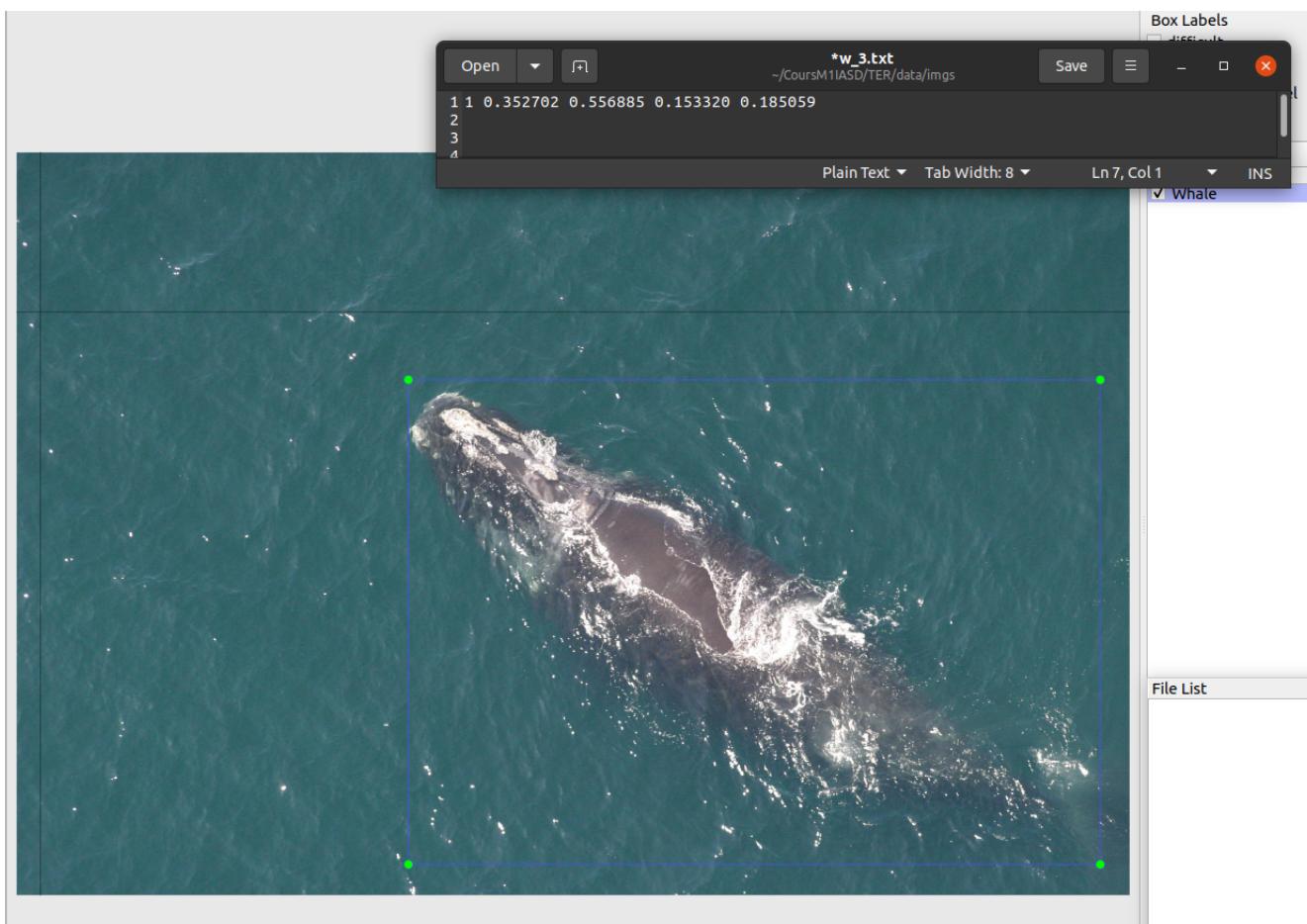


FIGURE 8 – LabelImg - Whole whale Box

2.3.1.2 Développement du premier modèle Voici une liste non exhaustive des paramètres modifiés sur le réseau.

listings

```
### Reglages parametres pour notre premier essai avec
500 imgs ###

!sed -i 's/batch=1/batch=64/' cfg/yolov3_training.cfg
# reduce subdivisions if you don't have enough RAM
!sed -i 's/subdivisions=1/subdivisions=32/' cfg/
yolov3_training.cfg
# change line max_batches to (classes*2000)
!sed -i 's/max_batches = 500200/max_batches = 4000/' cfg/
yolov3_training.cfg
# change line classes=80 to your number of objects in
each of 3 [yolo]-layers
!sed -i '610 s@classes=80@classes=10' cfg/
yolov3_training.cfg
!sed -i '696 s@classes=80@classes=10' cfg/
yolov3_training.cfg
!sed -i '783 s@classes=80@classes=10' cfg/
yolov3_training.cfg
# change [filters=255] to filters=(classes + 5)x3 in the
3 [convolutional] before each [yolo] layer
!sed -i '603 s@filters=255@filters=180' cfg/
yolov3_training.cfg
!sed -i '689 s@filters=255@filters=180' cfg/
yolov3_training.cfg
```

```
!sed -i '776 s@filters=255@filters=18@' cfg/  
yolov3_training.cfg
```

Plus de détail sur les réglages à effectuer ici⁴

2.3.1.3 Évaluation des résultats Malheureusement pour ce premier réseau nous avons perdu la trace de la métrique mAP durant l'entraînement, pour l'évaluation de nos résultats nous avons donc simplement regardé les sorties donné sur l'algorithme. Les sorties étaient plutôt bonnes même si le nombre de 500 images semblent un peu léger pour une détection optimale. ex de sortie après crop de l'image sur les box prédictes.

4. <https://github.com/AlexeyAB/darknet>



FIGURE 9 – exemple de sortie sur Yolov3

2.3.1.4 Problèmes et seconds prétraitements

Après, quelques recherches sur Internet sur les baleines noires on s'aperçoit que la caractéristique la plus distinctive d'une baleine noire se situe au niveau de sa tête. Les plaques rugueuses de la peau sur sa tête qui apparaissent blanches en raison du parasitisme par les poux de baleine. Il s'avère que cela les distingue non seulement des autres espèces de baleines, mais constitue également un bon moyen de différencier les individus. La solution présentée précédemment ne se concentre pas assez sur la tête mais considère la baleine toute la baleine. On va donc repeter le processus précédent mais en se concentrant uniquement sur la zone autour de la tête.

Nous constituons cette fois-ci un jeu de données d'entraînement de 3000 images.



FIGURE 10 – exemple de sortie sur Yolov3 avec seulement la tête

Après entraînement les résultats sont très bons, ci-dessous le graphique de l'entraînement n'indique pas de signe d'overfitting.

2.3.2 Head Aligner Yolo v4 et Algo d'Alignement

for training for both small and large objects use modified models :
=> use yolov4 which is really better

2.3.2.1 Premiers prétraitements

2.3.2.2 Développement du second modèle

2.3.2.3 Résultats

2.3.2.4 Alignement des images

2.3.3 Système de reconnaissance faciale

3 Conclusion

3.1 Perspectives d'amélioration

3.2 Ce qu'on a appris

Bibliographie

- [1] GREMM (baleinesendirect.org). Baleines noires de l'atlantique nord.
- [2] Robert Bogucki, Marek Cygan, Christin Brangwynne Khan, Maciej Klimek, Jan Kenty Milczek, and Marcin Mucha. Applying deep learning to right whale photo identification. *Conservation Biology*, 33(3) :676–684, 2019.