



UNIVERSITÉ DE  
**SHERBROOKE**

Élèves du deuxième cycle

Projet IFT-870

---

## **Identification de dauphins et de baleines à l'aide de réseaux convolutifs**

---

Firas ABOUDA *abof3001*

Gaétan REY *reyg3101*

Julien LEVARLET *levj2718*

Timothée WRIGHT *writ2801*

*Encadrant :*  
Mme. Nadia TAHIRI

# Table des matières

|   |          |
|---|----------|
| <b>Table des matières</b>                                     | <b>1</b> |
| <b>1 Nos Données</b>  | <b>3</b> |
| 1.1 Description du projet . . . . .                           | 3        |
| 1.1.1 Problématique . . . . .                                 | 3        |
| 1.1.2 Contexte . . . . .                                      | 3        |
| 1.2 Nos motivations pour le choix du sujet . . . . .          | 4        |
| 1.3 Historique des travaux en rapport avec le sujet . . . . . | 4        |
| 1.4 Description des données . . . . .                         | 4        |
| 1.5 Problèmes relatifs aux données . . . . .                  | 5        |
| 1.5.1 Taille des données . . . . .                            | 5        |
| 1.5.2 Résolution des images . . . . .                         | 5        |
| 1.5.3 Images polluées . . . . .                               | 6        |
| 1.5.4 Contraste et bruit . . . . .                            | 6        |
| 1.5.5 Distance photographe - animal . . . . .                 | 7        |
| 1.5.6 Multiples animaux . . . . .                             | 7        |
| 1.5.7 Base de données fortement débalancée . . . . .          | 8        |
| 1.5.8 Résumé . . . . .  | 8        |
| <b>2 Algorithme</b>   | <b>9</b> |
| 2.1 Notre analyse du sujet . . . . .                          | 9        |
| 2.1.1 Idée de traitement du sujet . . . . .                   | 9        |
| 2.1.2 Outils . . . . .  | 9        |
| 2.2 Prétraitements . . . . .                                  | 10       |
| 2.2.1 Augmenter nos données . . . . .                         | 10       |
| 2.2.2 Équilibrer la base de donnée . . . . .                  | 10       |
| 2.2.3 Redimensionnement des images . . . . .                  | 10       |
| 2.2.4 Fausses bonnes idées . . . . .                          | 11       |
| 2.3 Isolation du sujet de l'image . . . . .                   | 12       |
| 2.3.1 Rognage des images . . . . .                            | 12       |

|       |  |    |
|-------|--|----|
| 2.3.2 | Masque de segmentation . . . . .                     | 13 |
| 2.4   | Outils de classification . . . . .                   | 14 |
| 2.4.1 | Architectures de CNN . . . . .                       | 14 |
| 2.4.2 | Entrainement . . . . .                               | 14 |
| 2.4.3 | Fine tuning . . . . .                                | 15 |
| 2.4.4 | Les Swin transformers, alternative aux CNN . . . . . | 15 |
| 2.5   | Comparaison d'individus . . . . .                    | 15 |
| 2.5.1 | Embedding . . . . .                                  | 16 |
| 2.5.2 | Arcface . . . . .                                    | 16 |
| 2.6   | Utilisation des GPU et des TPU . . . . .             | 16 |
| 2.7   | Conclusion et application . . . . .                  | 17 |

# 1 Nos Données

## 1.1 Description du projet

### 1.1.1 Problématique

Nous utilisons les empreintes digitales et la reconnaissance faciale pour identifier les personnes, mais pouvons-nous utiliser des approches similaires avec les mammifères marins ?

### 1.1.2 Contexte

Les chercheurs suivent manuellement la vie marine par la forme et les marques sur leurs queues, leurs nageoires dorsales, leurs têtes et autres parties du corps des mammifères marins. L'identification par des marques naturelles via des photographies, appelée photo-identification, est un outil puissant pour la science des mammifères marins. Il permet de suivre les animaux individuels au fil du temps et il permet d'évaluer l'état et les tendances de la population.

L'objectif de ce projet est d'automatiser la photo-identification des baleines et des dauphins, les chercheurs pensent pouvoir réduire les temps d'identification des images de dauphins et de baleines de plus de 99% grâce au machine learning.

Actuellement, la plupart des instituts de recherche s'appuient sur une mise en correspondance manuelle, chronophage et parfois imprécise par l'œil humain. Des milliers d'heures sont consacrées à l'appariement manuel, qui consiste à regarder des photos pour comparer un individu à un autre, trouver des correspondances et identifier de nouveaux individus.

Les algorithmes développés dans le cadre de ce concours seront mis en œuvre dans Happywhale, une plateforme Web de collaboration, de recherche et de science citoyenne. Sa mission est d'accroître la compréhension globale et de prendre soin des environnements marins grâce à une science et une éducation de haute qualité en matière de conservation. Happywhale vise à rendre facile et enrichissante pour le public de participer à la science en créant des outils innovants pour impliquer toute personne intéressée par les mammifères marins. La plateforme est également au service de la communauté de la recherche avec de puissants outils collaboratifs.

Dans cette compétition, nous devons développer un modèle pour faire correspondre les baleines et les dauphins par des caractéristiques uniques, mais souvent subtiles, de leurs marques naturelles. Nous devons porter une attention particulière aux nageoires dorsales et aux vues latérales du corps dans des ensembles d'images provenant d'un ensemble de données multi-espèces construit par 28 instituts de recherche. L'objectif est de créer un modèle à la fois rapide et précis.

## 1.2 Nos motivations pour le choix du sujet

Nous souhaitions quelque chose de plus complexe qu'un projet de classification avec des données textuelles dans un tableau de type CSV. Nous avions envie de mettre à profit nos connaissances en réseau de convolution et en forage de données dans un projet ambitieux et d'actualité. Nous avons ainsi cherché des projets qui utilisaient des données de types images ou vidéo et nous nous sommes mis d'accord sur ce projet. De plus, le fait que le projet soit un challenge en cours est très intéressant, car tous les jours de nouvelles discussions sur le sujet apparaissent et nous pouvons nous comparer en temps réel aux autres participants.

## 1.3 Historique des travaux en rapport avec le sujet

Lors de la précédente compétition HappyWhale [1], la tâche consistait à prédire les baleines à bosse individuelles à partir d'images de leurs douches. Les baleines et les dauphins de cet ensemble de données pouvaient être identifiés par des formes, des caractéristiques et des marques (certaines naturelles, d'autres acquises) des nageoires dorsales, du dos, de la tête et des flancs :

Après des siècles d'intense chasse à la baleine, les populations de baleines en rétablissement ont encore du mal à s'adapter au réchauffement des océans et luttent chaque jour pour concurrencer l'industrie de la pêche industrielle pour se nourrir.

Pour aider les efforts de conservation des baleines, les scientifiques utilisent des systèmes de surveillance photographique pour surveiller l'activité océanique. Ils utilisent la forme des queues des baleines et des marques uniques trouvées dans les images pour identifier les espèces de baleines qu'ils analysent et enregistrent méticuleusement la dynamique et les mouvements des goussettes de baleines. Au cours des 40 dernières années, la plupart de ce travail a été effectué manuellement par des scientifiques individuels, laissant une énorme mine de données inexploitées et sous-utilisées.

Dans ce concours, nous sommes mis au défi de créer un algorithme pour identifier individuellement les baleines et les dauphins dans les images. Nous analyseront la base de données de Happywhale de plus de 25 000 images, recueillies auprès d'institutions de recherche et de contributeurs publics. Les contributions ont pour but d'ouvrir de riches champs de compréhension pour la dynamique des populations de mammifères marins dans le monde entier.

Certaines espèces et certains individus ont des caractéristiques très distinctes, d'autres sont beaucoup moins distinctes. De plus, les caractéristiques individuelles peuvent changer au fil du temps.

Ce concours élargit considérablement cette tâche : les données de ce concours contiennent des images de plus de 15 000 mammifères marins uniques appartenant à 30 espèces différentes, collectées auprès de 28 organismes de recherche différents. Les personnes ont été identifiées manuellement et ont reçu un identifiant individuel par des recherches marines et notre tâche est d'identifier correctement ces individus dans les images.

## 1.4 Description des données

Les données de ce concours contiennent des images de plus de 15 000 mammifères marins uniques appartenant à 30 espèces différentes, collectées auprès de 28 organismes de recherche différents. Les données sont organisées dans des dossiers et des fichiers reliés par les identifiants de nos espèces.

- train\_images/ est le dossier contenant les images d’entraînement
- train.csv fournit l’espèce et l’identifiant individuel pour chacune des images de dauphin
- test\_images/ est un dossier contenant les images de test, donc non labellisées
- sample\_submission.csv est un exemple de fichier de soumission au format correct.

| <a href="#">▲ image</a>         | <a href="#">▲ espèce</a>                                | <a href="#">▲ identifiant_individuel</a> |
|---------------------------------|---|--|
| <b>51033</b><br>valeurs uniques | gros nez_dauphin 19%<br>béluga 15%<br>Autre (33926) 66% | <b>15587</b><br>valeurs uniques          |
| 00021adfb725ed.jpg              | baleine à tête de melon                                 | cadddb1636b9                             |
| 000562241d384d.jpg              | baleine à bosse   | 1a71fbb72250                             |
| 0007c33415ce37.jpg              | false_killer_whale                                      | 60008f293a2b                             |

FIGURE 1.1 – Fichier CSV de la compétition contenant les données entraînement qui relie individus, espèces et images

Pour chaque image, notre tâche est de prédire l’identifiant individuel de chaque dauphin, aucune information sur les espèces n’est à prédire. Il y a des individus dans les données de test qui ne sont pas représentés dans les données d’entraînement, ces images devront donc être prédites comme nouveaux individus.

## 1.5 Problèmes relatifs aux données

### 1.5.1 Taille des données

Une des difficultés de ce projet vient de la masse des données, celle-ci prennent plus de 60 gigaoctets au format zip donc nous n’avons pas la place suffisante pour télécharger le dossier sur nos ordinateurs. En contre-partie, nous avons beaucoup de données pour entraîner au mieux notre modèle.

Pour pallier ce problème, nous utiliserons donc l’exécution de notebook dans le cloud de Kaggle [2] pour entraîner nos modèles, tout en faisant des tests en local sur de petits échantillons de la base de données.

### 1.5.2 Résolution des images

Les images proviennent de sources différentes et n’ont pas toutes les mêmes dimensions, ce qui rend leur manipulation compliquée en l’absence de prétraitements adéquats.

### 1.5.3 Images polluées

Dans la base de données, certaines images ne contiennent pas uniquement la photo prise, mais aussi des éléments ajoutés par la suite comme des annotations, ou encore une capture d'écran d'un lecteur vidéo.

Ces informations supplémentaires ne doivent pas être prises en compte, car un algorithme qui apprend à lire les annotations ne sera pas performant des images dont l'identifiant du dauphin est inconnu et cela serait considéré comme du sur-apprentissage.

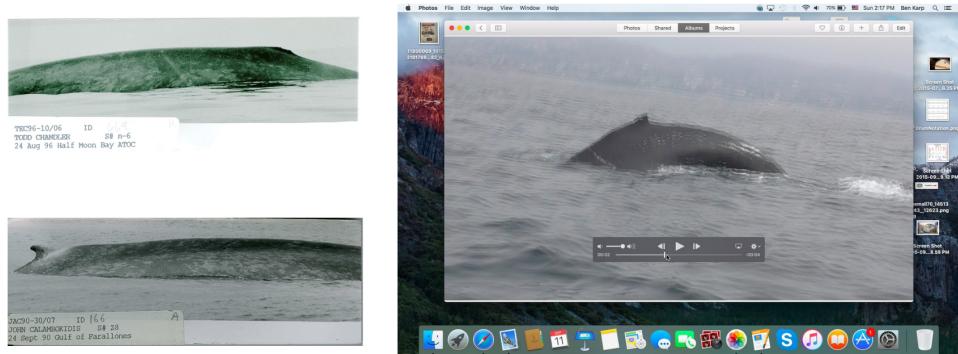


FIGURE 1.2 – Exemple d'éléments ajoutés à l'image d'origine

Source : <https://www.kaggle.com/c/happy-whale-and-dolphin/discussion/308026>

### 1.5.4 Contraste et bruit



FIGURE 1.3 – Exemple d'images prises dans des conditions climatiques différentes

La première image de la figure 1.3 semble en nuance de gris, c'est l'éclairage qui donne cette impression. Sur la seconde au contraire, le fond est d'un bleu beaucoup plus prononcé. Ainsi l'éclairage a un impact sur notre capacité à distinguer l'aileron du fond de l'image, c'est-à-dire de l'eau, du ciel ou de la banquise.

De plus sur la première image, nous pouvons remarquer que l'aileron est vraiment petit au milieu à gauche de l'image et à l'œil, il est difficile d'identifier des marques qui permettent de distinguer ce dauphin d'un autre.

Sur le second dauphin l'aileron est bien au centre de l'image visible et au centre, mais il y a un petit effet d'ombre qui le rend plus sombre que le reste de l'image. On remarque aussi des jeux d'ombres et de lumière à cause du soleil.

L'océan peut également apparaître de couleurs différentes, allant du gris au rose en passant par le vert et toutes les teintes de bleu. La question du fond de la photo va donc se poser.

### 1.5.5 Distance photographe - animal

Certaines images sont prises de loin et les baleines sont peu ou pas reconnaissables.

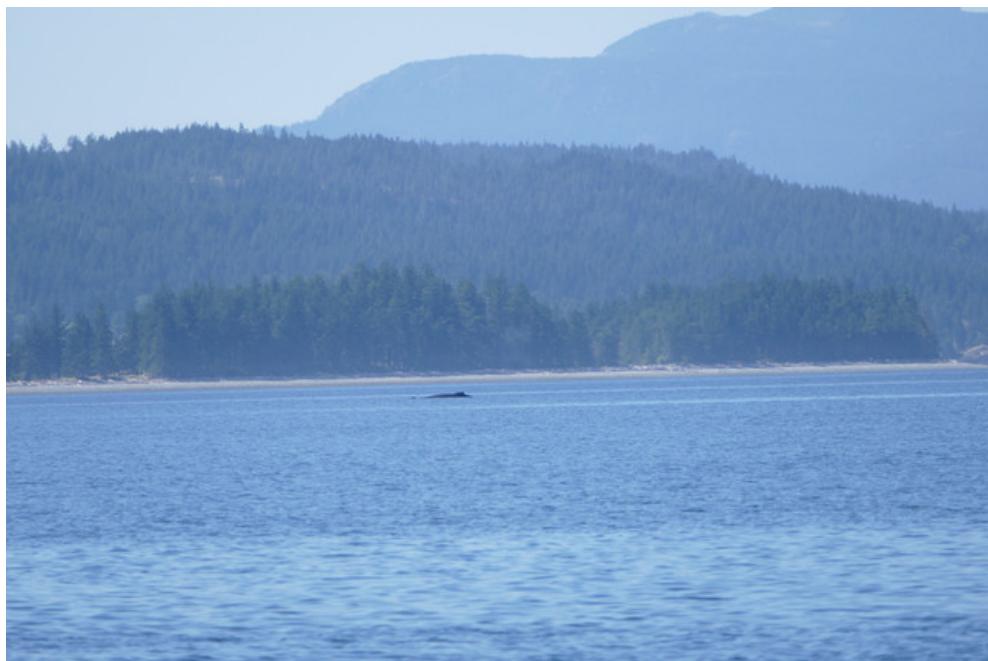


FIGURE 1.4 – baleine très éloignée

Pour ces dernières, il faut soit réaliser de très gros traitements, soit les identifier comme difficile à traiter et les éliminer, en effet même en réussissant à isoler le mammifère de l'image, la résolution pourrait être trop faible pour distinguer les caractéristiques nécessaires à son identification.

### 1.5.6 Multiples animaux

Nous devons détecter un animal particulier par image. Cependant, plusieurs photos contiennent plusieurs animaux. Parfois une baleine et un dauphin, mais également des bancs de dauphins.



FIGURE 1.5 – Exemples de photos avec plusieurs animaux.

Source : <https://www.kaggle.com/c/happy-whale-and-dolphin/discussion/308026>

Lors de nos prétraitements, il faudra donc faire particulièrement attention à ce que nos algorithmes ne porte pas l'attention sur un dauphin en retrait dans l'image, mais plutôt toujours se focaliser sur l'animal en premier plan, qui est sûrement le dauphin qui a été annoté manuellement par un humain.

Dans un premier temps, on pourra également se passer de ces données là qui peuvent être dures à traiter.

### 1.5.7 Base de données fortement débalancée

Le nombre d'images est très différent pour chaque individu. Certains individus ne sont représentés qu'une seule fois, alors que d'autres peuvent être présent une vingtaine de fois.

Au moment de mesurer les performances de nos algorithmes, il faudra donc être particulièrement attentif à ne pas uniquement s'intéresser à la justesse globale, mais aussi vérifier que les individus même peu présent peuvent être tout de même identifiés.

Des prétraitements pourront également être réalisés pour compenser cela.

Source : <https://www.kaggle.com/c/happy-whale-and-dolphin/discussion/308547>

### 1.5.8 Résumé

Nous pouvons donc voir que les images sont toutes très différentes les unes des autres à la fois à cause des conditions d'éclairage et de la météo. Mais aussi à cause des conditions dans lesquelles la photo a été prise :

1. la proximité avec le dauphin
2. la technique : zoom, panorama
3. la taille de l'image selon l'angle et l'appareil
4. le cadrage : l'aileron peut être au centre de l'image ou excentré
5. la présence d'autres animaux
6. la présence d'annotation

En conséquence, il faudra effectuer des prétraitements pour homogénéiser nos images et éviter que ces différences influent lors l'apprentissage.

## 2 Algorithm

Notre sujet consiste donc à faire de la reconnaissance de formes sur des images. Pour cela, les réseaux de neurones convolutifs ou CNN (Convolutional Neural Network) sont parfaitement adaptés.

Nous parlerons dans ce chapitre des différents prétraitements nécessaire à appliquer à nos données pour rendre efficace l'utilisation des CNN puis nous détaillerons les différents outils d'apprentissage utilisables dans ce contexte.

### 2.1 Notre analyse du sujet

#### 2.1.1 Idée de traitement du sujet

Notre objectif de reconnaissance peut être séparé en deux étapes, dans un premier temps, il est utile de localiser plus ou moins précisément le sujet qui nous intéresse dans l'image. Dans un second temps, en ayant isolé l'objet de l'image, nous pouvons passer à la classification.

Notre base de données contient plus de 15000 mammifères avec de nombreuses images pour chacun d'entre eux, le tout dépasse les 60 giga-octets de données. Pour pouvoir réaliser de nombreux tests, appliquer et comparer les techniques mentionnées dans ce chapitre, nous allons devoir subdiviser nos données.

Nous avons décidé de nous baser dans un premier temps sur une vingtaine d'individus plus ou moins représentés pour avoir un échantillon d'entraînement représentatif. Par ailleurs, nous subdiviserons cet échantillon en 80% de données d'entraînement et 20% de données de test en équilibrant les classes pour valider nos modèles.

Si les résultats sont concluants alors seulement nous essayerons d'entraîner un modèle sur un plus grand échantillon des données, utiliser un plus grand nombre de classes dès le début serait une grosse perte de temps.

#### 2.1.2 Outils

Pour implémenter nos algorithmes d'apprentissage, nous utiliseront l'API de Pytorch [3]. Cette API est la plus utilisée dans les challenges récents (avec Tensorflow) et permet l'implémentation de réseaux neuronaux ainsi que certains prétraitements.

Pytorch [3] est notamment adapté pour l'accélération du calcul par GPU et TPU dans certains cas. Cela sera donc parfaitement adapté à l'exécution cloud proposée par Kaggle [2], allouant des GPU et TPU (pour une durée limitée pour ces derniers).

Dans le cas des traitements sur les images, la librairie Python la plus utilisée est OpenCV [4]. Elle permet d'effectuer la plupart des traitements sur les images qui pourront être utiles à notre projet.

## 2.2 Prétraitements

### 2.2.1 Augmenter nos données

Comme on l'a remarqué précédemment, certains individus ne sont présents qu'une seule fois dans la base de données. Il est donc très utile de créer manuellement de nouvelles données d'un même individu.

Plusieurs techniques sont utilisées pour cela. Il est notamment possible de superposer du bruit moyen aux images pour augmenter le nombre de d'images par individus et surtout diminuer l'influence du contexte, luminosité, éclairage, cadrage. Mais on peut également ajouter du flou cinétique, passer l'image en noir et blanc, ou effectuer des petites rotations (mais pas une réflexion géométrique, comme on le verra plus tard dans les fausses bonnes idées).

Toutes les équipes gagnantes du précédent challenge d'Happywhale ont utilisé l'augmentation de données.

### 2.2.2 Équilibrer la base de donnée

La base de données étant très fortement déséquilibrée, il ne sera pas possible d'avoir le même nombre d'images pour chaque individu, mais grâce à l'augmentation des données on peut s'assurer d'avoir des images pour chaque individu à la fois dans les données de test et d'entraînement.

Dans le but de garder la même répartition des images des individus, nous utiliserons la stratification lors de la séparation de nos ensembles.

Cette méthode permet lors de la séparation de notre base de données (par exemple entre ensemble d'entraînement et de test) de faire la séparation selon la même proportion (par exemple 80% entraînement, 20% test) pour chaque classe.

### 2.2.3 Redimensionnement des images

Nos images sont toutes de tailles très différentes, il est donc nécessaire de les normaliser pour pouvoir les utiliser efficacement. Une première idée serait de prendre une taille fixe et redimensionner toutes nos images dans ce format.

Pour redimensionner des images, il existe deux principales méthodes :

- Le redimensionnement qui "aplatit" l'image
- L'ajout de padding (bandes noires) pour garder le ratio de l'image de départ

Cependant en faisant comme dans le premier cas, ceci risque de générer des déformations et perdre en informations. C'est pourquoi le padding peut être préférable pour obtenir des images de tailles carrées avant de redimensionner.

Mais il semble que dans les participants au challenge, beaucoup préfèrent utiliser un redimensionnement classique, quitte à obtenir des images aplatis. Un exemple est présent ici : <https://www.kaggle.com/c/happy-whale-and-dolphin/discussion/304686>

Ensuite, là encore il faut bien choisir la nouvelle taille, si elle est trop petite nous perdrions trop d'information, si elle est trop grande le temps de calcul sera trop long.

Pour pouvoir faire plusieurs tests rapides, nous commencerons par essayer sur une dimension 128\*128 avant de monter vers des dimensions plus grande 256\*256 puis 512\*512 pour vérifier si nos résultats sont meilleurs en gardant l'information le plus intact possible.

Pour redimensionner nos images, on peut utiliser des méthodes d'interpolation telles que INTER\_CUBIC d'OpenCV [4] qui réalise une interpolation bicubique.

Cependant, cette méthode de redimensionnement ne permet pas de cibler l'information utile, c'est juste une normalisation de la taille des images, pour gagner en performance, il sera peut-être plus intéressant de cibler la partie qui représente le dauphin au préalable et ne donner que cette partie à notre réseau.

#### 2.2.4 Fausses bonnes idées

Certaines idées listées dans cette section peuvent sembler intéressantes, mais sont en réalité inutiles, voire néfastes.

##### Modifier le contraste

Sur certaines images, en modifiant le contraste, on voit apparaître à l'œil nu des détails que l'on ne voyait pas avant.

Exemple :

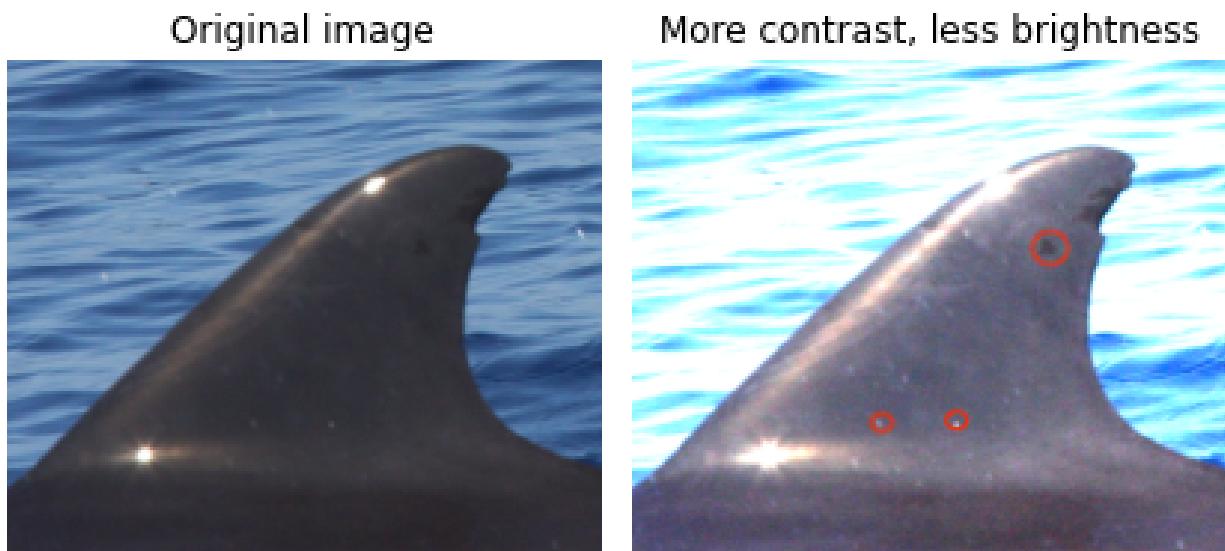


FIGURE 2.1 – Augmentation du contraste

Cependant, en modifiant le contraste, on détruit de l'information. De plus, on préfère en vision par ordinateur utiliser des réseaux de neurones à convolutions pour apprendre le traitement adapté et extraire les patrons utiles plutôt qu'essayer de rendre ces patrons plus visibles. S'il faut augmenter le contraste pour observer un détail, alors la première couche de convolution agira sans doute comme un rehausseur de contraste.

## Retourner les images

En vision par ordinateur, pour augmenter la quantité de données et pour aider à la reconnaissance d'un objet dans différents contextes, on utilise souvent le symétrique des images.

Dans ce contexte, cela peut être une mauvaise idée. Par exemple, si on identifie les dauphins à partir de taches sur leurs ailerons, alors on pourrait voir une tâche sur un aileron dans un sens, mais pas dans l'autre.

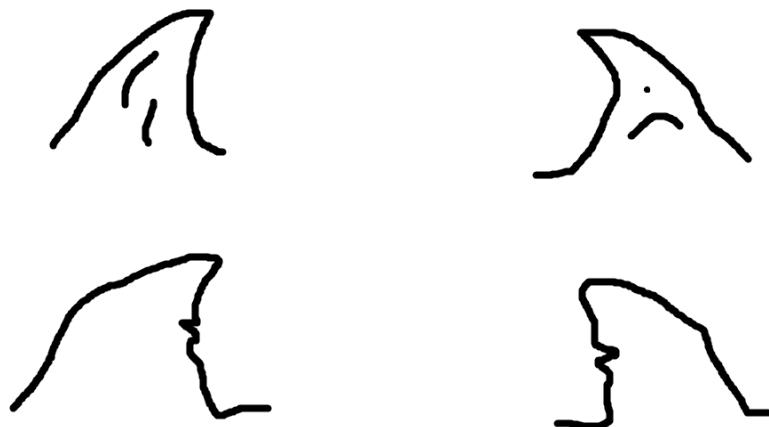


FIGURE 2.2 – Exemple de vision des éléments caractéristiques d'un aileron

Sur le schéma ci-dessus, en haut l'aileron à un schéma différent sur ses différents flancs. Effectuer une symétrie fera donc croire à un nouvel individu si l'on s'intéresse aux motifs de l'aileron.

Par contre, si l'on s'intéresse à la forme de l'aileron, ou à une cicatrice comme c'est le cas sur le bas du schéma, alors le motif sera présent après une réflexion de l'image et cette opération peut être intéressante pour augmenter les données.

## 2.3 Isolation du sujet de l'image

### 2.3.1 Rognage des images

#### Qu'est-ce que le rognage

Le rognage ou cropping en anglais est le fait de sélectionner une partie de l'image. Cela permet de n'avoir sur l'image que le sujet qui nous intéresse. Les deux avantages de ces méthodes sont de diminuer le temps de calcul en passant une image plus petite à notre réseau et éviter que celui-ci apprenne sur des éléments du contexte qui ne nous intéresse pas comme un bateau ou un autre mammifère.

#### Rognage par bounding box

Les Bounding boxes ou "boîtes englobantes" en français sont des rectangles qui marquent des objets sur une image. Pour trouver ces bounding box des détecteurs existent tels que YOLO ou Detic. Une description de l'outil YOLO est donnée dans la section suivante.

Une fois la détection d'un objet faite, on peut agrandir la sélection (pour être sûr de ne louper aucune partie de l'image) et redimensionner l'image selon cette nouvelle taille. Outre quelques erreurs, on aura alors des images avec l'animal ou son aileron bien visible et prenant la plus grande partie de l'image.

## YOLO

YOLO [5] est l'acronyme de You Only Look Once (vous ne regardez qu'une fois). Il s'agit d'un cadre de détection d'objets extrêmement rapide utilisant un seul réseau convolutif. YOLO est souvent plus rapide que les autres systèmes de détection d'objets parce qu'il examine l'ensemble de l'image en une seule fois, au lieu de la balayer pixel par pixel.

Contrairement à l'approche traditionnelle où l'on commence par détecter les régions intéressantes puis on réalise de la classification sur la région intéressante. Yolo prédit toutes les boites en une seule fois.

Il existe aussi l'alternative SSD pour single shot detector et Detic pour effectuer le même traitement. Ces outils sont utilisés par plusieurs équipes de Kaggle [2], nous pourrons les comparer dans notre implémentation.

Avec Pytorch[3] et Yolo, il est très facile d'obtenir des bounding box. Reste à savoir si ces box sont intéressantes et si cela fonctionne bien sur des données brutes.

Un exemple d'utilisation est disponible ici :

<https://www.kaggle.com/awesaf49/happywhale-cropped-dataset-yolov5/>

## Applications

Une idée serait donc d'utiliser la détection de bounding box pour rogner nos images avant de les redimensionner dans le but d'entraîner nos réseaux de convolution.

### 2.3.2 Masque de segmentation

Certains compétiteurs comme Remek Kinas [6] ont essayé d'isoler l'aileron du dauphin du fond de l'image, il reste cependant à voir si c'est vraiment utile et si cela ne nous fait pas perdre de l'information au niveau du contour de la nageoire dorsale du dauphin qui nous intéresse tout particulièrement.

Visuellement le résultat est bon, reste à savoir si pour l'ordinateur, c'est plus facile à utiliser ou s'il y a des pertes d'informations. Nous essayerons cette technique pour s'en assurer.

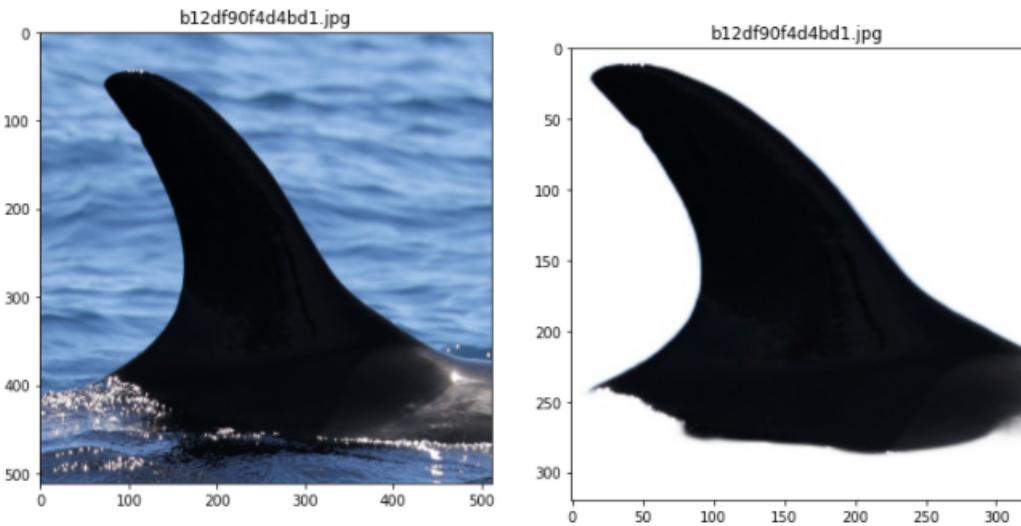


FIGURE 2.3 – Aileron d'un dauphin avant et après isolation du fond

## 2.4 Outils de classification

### 2.4.1 Architectures de CNN

Il existe des dizaines et des dizaines d'architectures de CNN. De manière générale, plus elles sont récentes, meilleurs sont les résultats. Mais elles peuvent également répondre à un besoin particulier.

Ces architectures étaient généralement dévoilées et reconnues lors de la compétition annuelle "ImageNet Large Scale Visual Recognition Challenge" (ILSVRC) réalisée sur la célèbre base de données d'image ImageNet.

Parmi les architectures les plus célèbres, on retrouve AlexNet, le gagnant en 2012, ZFNet en 2013, GoogLeNet et VGGnet en 2014 ou encore ResNet en 2015.

Toutes ces architectures peuvent être intéressantes pour notre problème.

Plusieurs personnes semblent utiliser Effnet, une architecture plus récente qui a notamment obtenu la première place du Google landmark recognition en 2020 (dont on parlera plus tard).

Pour la compétition précédente d'Happywhale (2018), les modèles les plus utilisés par les 10 meilleures équipes étaient : ResNet, DenseNet et ResNeXt.

### 2.4.2 Entrainement

Pour l'entraînement, il est possible de faire de la recherche d'hyperparamètres, en particulier sur le taux d'apprentissage.

En plus de cela, il est possible d'utiliser des substituts à la descente de gradient. En particulier, l'optimizer "Adam" a été utilisé dans plusieurs challenges de ce type par les équipes ayant obtenu les meilleurs résultats.

La descente de gradient stochastique maintient un taux d'apprentissage unique (appelé alpha) pour toutes les mises à jour de poids et le taux d'apprentissage ne change pas pendant l'entraînement.

Adam combine les avantages de deux extensions de la descente de gradient qui maintiennent un taux d'apprentissage par paramètre, à savoir AdaGrad qui améliore les performances sur les problèmes avec des gradients dispersés et RMSProp qui est adapté pour les problèmes ayant du bruit.

### 2.4.3 Fine tuning

Le fine tuning consiste à réutiliser les poids d'un réseau déjà entraîné et le réutiliser comme initialisation pour un autre problème du même domaine.

Il existe ensuite plusieurs stratégies telles que réutiliser tout le réseau de neurones ou enlever certaines couches.

Cela permet d'accélérer l'entraînement et de palier certains problèmes dus à la taille du dataset.

Cela pourrait donc être très utile dans notre cas, mais nous ne disposons pas d'un tel réseau. On peut cependant se poser la question si ça vaut la peine d'entraîner au préalable un réseau sur une base de données plus vaste d'animaux marins.

### 2.4.4 Les Swin transformers, alternative aux CNN

Fin 2020, un papier a présenté une alternative aux CNN appelée swin pour Shifted Windows.

Nous avons parlé de la compétition ILSVRC. Au cours des dernières années, les résultats devaient tous très bons et ce problème a été plus ou moins défini comme "résolu". Une nouvelle compétition a alors émergé : la "Google's Landmark Recognition Kaggle Competition". Le but ? Reconnaître les coordonnées des points d'intérêts (ou landmark) sur une base de donnée de plus de 4 millions d'images.

Et lors de l'édition 2021 de cette compétition, ce sont les swin qui ont remporté la mise.

Cependant, les transformers prennent plus de temps à converger que les CNN ce qui rend leur implémentation compliquée pour nous. De plus, il s'agit d'un champ de recherche peut-être trop récent pour nous qui ne cherchons pas forcément à avoir la meilleure solution, mais une bonne solution que nous comprenons.

## 2.5 Comparaison d'individus

Ce sujet est composé de deux problématiques. La première est la reconnaissance d'un dauphin ou d'une baleine sur une image. La deuxième est l'identification unique de l'animal.

Pour cette deuxième problématique, il s'agit d'être capable, à partir de deux images de dauphins, de dire s'il s'agit du même individu.

Pour cela, on peut s'appuyer sur des algorithmes existants pour l'identification unique d'êtres humains.

### 2.5.1 Embedding

Lors de l'utilisation d'un CNN pour la classification, on utilise un réseau de neurones pour prédire une classe. Cependant, lorsqu'on compare deux nouveaux individus et l'on cherche à savoir s'ils sont proches, on ne peut pas simplement comparer les classes.

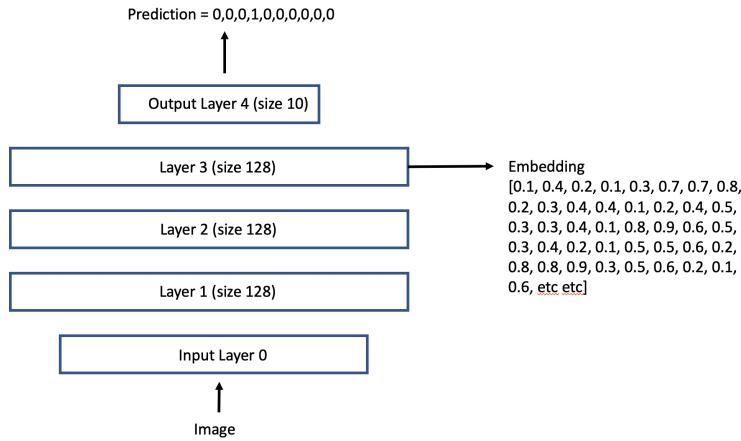


FIGURE 2.4 – Exemple d'un CNN

On aimerait pouvoir comparer les individus en utilisant le vecteur de la dernière couche d'activation. Ce vecteur est utile puisque représentant les patterns de l'image détectés sur l'image. Pour comparer les vecteurs on utilise une mesure de distance telle que la distance cosinus. Cependant, il n'y a aucune raison que les animaux proches ait des vecteurs proches car le réseau n'a pas été entraîné pour ça.

### 2.5.2 Arcface

ArcFace ajoute une loss à l'entraînement pour encourager les embeddings des classes proches à être similaires.

ArcFace est donc un modèle d'apprentissage automatique qui permet de prendre 2 photos de visages et retourne la probabilité qu'il s'agisse de la même personne.

Plusieurs équipes utilisent ArcFace pour ce challenge afin de déterminer la probabilité que deux images de dauphins montrent le même individu. De plus, dans la dernière compétition d'Happywhale la plupart des meilleures équipes utilisaient ArcFace.

Arcface offre une mesure d'erreur (Arface loss) que l'on pourra utiliser, mais ce n'est pas la seule façon de comparer deux individus. Certaines personnes ont obtenu de très bons résultats en utilisant d'autres mesures de similarité.

## 2.6 Utilisation des GPU et des TPU

Dans un ordinateur, le processeur ou CPU gère toute la logique, les calculs et les entrées/sorties de l'ordinateur.

Le processeur graphique ou GPU, permet d'assister le CPU dans les fonctions de calcul d'image, d'affichage à l'écran ou d'écriture sur mémoire de masse. Un processeur graphique a généralement une structure hautement parallèle qui le rend efficace pour une large palette de tâches graphiques.

L'unité de traitement de tenseur ou TPU est un circuit intégré développé par Google spécifiquement pour accélérer les algorithmes et calculs en intelligence artificielle. Les TPU sont développés spécifiquement pour Tensorflow, qui appartient à Google.

Cet article sur towardsdatascience [7] compare les performances des CPU, GPU et TPU. Les résultats semblent indiquer que les TPU permettent de multiplier la vitesse de traitement par 100 comparés à un CPU et par 3.5 comparé à un GPU lorsqu'entrainé sur un modèle Xception.

Sur de nombreux problèmes où le GPU venait à manquer de mémoire, le TPU obtenait des résultats assez rapidement.

Cependant, pour des batchs assez petits ou de petites prédictions, l'expérimentation montre que les TPU allaient 3x plus vite que les CPU, mais 3x moins vite que les GPU.

Kaggle [2] offre la possibilité d'utiliser leurs GPUs Nvidia K80 pendant une durée de 30 heures par semaine, voire plus dépendamment de l'utilisation du service.

Il est aussi possible d'utiliser leurs TPU pour un maximum de 20 heures chaque semaine.

Travaillant avec pytorch[3] nous n'utiliserons pas les TPU. L'utilisation de GPU sera de toute façon suffisante car, étant donné le temps consacré à ce projet, nos prétraitements et notre modèle ne seront pas parfait. Il est également important de rappeler que le coût à l'heure d'un TPU est 5 fois plus important qu'un GPU.

## 2.7 Conclusion et application

Dans le cadre de ce projet, nous prévoyons dans un premier temps d'entrainer nos algorithmes sur un sous-ensemble du dataset. Pour cela nous sélectionneront les individus dont nous avons un certain nombre de photos, par exemple minimum 5 pour commencer.

Nous utiliserons également des images redimensionnées. Pour cela, nous rognerons probablement l'image, pour ne garder que la partie importante via des bounding box ou des masques de segmentation.

Ces images rognées seront redimensionnées dans des tailles qui varieront en fonction de la performance de notre méthode.

Nous pourrons ensuite augmenter nos données.

Une fois le dataset nettoyé et augmenté obtenu, nous entrainerons nos algorithmes de classification dessus.

Il est probable que nous nous contentions d'architectures simples comme le ResNet ou le DenseNet, même si nous en essayerons plusieurs. Concernant la fonction de coût, nous utiliserons peut-être un softmax dans un premier temps pour une question de simplicité puis nous essaierons d'utiliser arcface.

Une fois que cela fonctionnera, on pourra essayer d'améliorer notre modèle en utilisant par exemple l'optimiseur Adam.

Nous prévoyons d'utiliser les API pytorch [3] et OpenCV [4] pour nous permettre d'implémenter nos méthodes.

Au vu de la taille de la base de données, nos machines personnelles ne sont pas suffisantes pour faire toutes les étapes précédemment énumérées, nous ferons donc des tests en local sur des

échantillons, pour valider que notre code fonctionne. Dans un second temps, nous utiliserons l'exécution cloud de Kaggle, pour un entraînement plus conséquent sur plus de données.

## Liens cités dans ce rapport

<https://www.kaggle.com/awsaf49/happywhale-cropped-dataset-yolov5/>, 13  
<https://www.kaggle.com/c/happy-whale-and-dolphin/discussion/304686>, 10  
<https://www.kaggle.com/c/happy-whale-and-dolphin/discussion/308026>, 6, 8  
<https://www.kaggle.com/c/happy-whale-and-dolphin/discussion/308547>, 8

## Bibliographie

- [1] HAPPYWHALE, *Humpback whale identification*, <https://www.kaggle.com/c/humpback-whale-identification>, 2019.
- [2] Kaggle, plateforme web de science des données, <https://kaggle.com>.
- [3] A. PASZKE, S. GROSS, F. MASSA et al., « PyTorch : An Imperative Style, High-Performance Deep Learning Library, » in *Advances in Neural Information Processing Systems 32*, H. WALLACH, H. LAROCHELLE, A. BEYGELZIMER, F. d'ALCHÉ-BUC, E. Fox et R. GARNETT, éd., Curran Associates, Inc., 2019, p. 8024-8035. adresse : <http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf>.
- [4] G. BRADSKI, « The OpenCV Library, » *Dr. Dobb's Journal of Software Tools*, 2000.
- [5] J. REDMON, S. K. DIVVALA, R. B. GIRSHICK et A. FARHADI, « You Only Look Once : Unified, Real-Time Object Detection, » *CoRR*, t. abs/1506.02640, 2015. arXiv : 1506.02640. adresse : <http://arxiv.org/abs/1506.02640>.
- [6] R. KINAS, *Isolation des ailerons*, <https://www.kaggle.com/remekkinas/whales-feature-matching-loftr-kornia>, 2022.
- [7] P. MOONEY, *When to use CPUs vs GPUs vs TPUs in a Kaggle Competition ?* <https://towardsdatascience.com/when-to-use-cpus-vs-gpus-vs-tpus-in-a-kaggle-competition-9af708a8c3eb>, 2020.