Neural Computing and Deep Learning

2019 MOD006568 TRI2 F01CAM

Element 010 - Component 2: Report

Right Whale Recognition

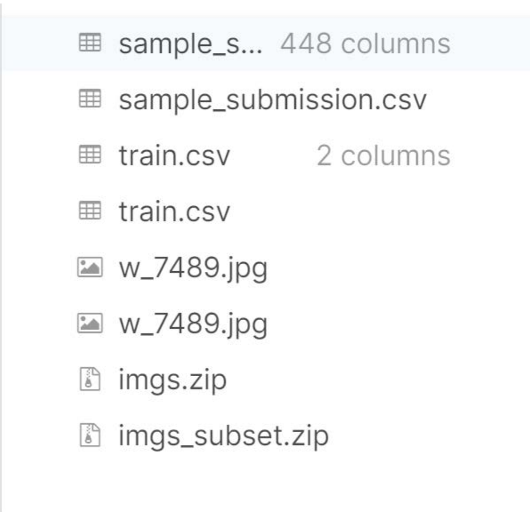
SID: 0925739

Table of Contents

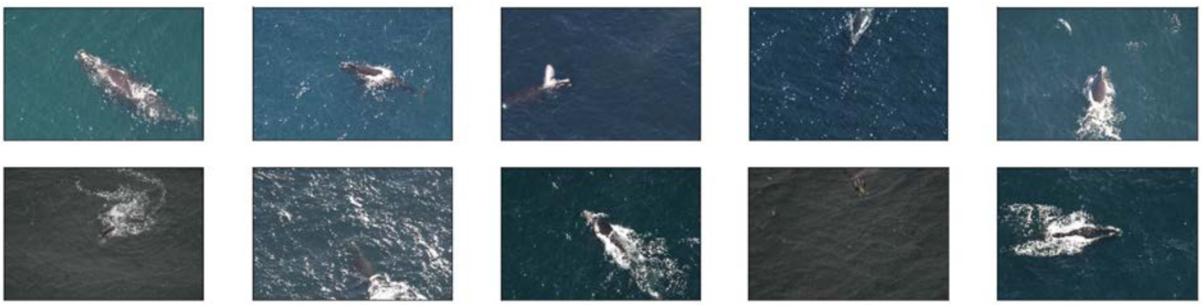
Background

The “Right Whale Recognition is a Kaggle Image Recognition Challenge competition to create a Machine Learning model to classify the endangered North Atlantic right whales in their respective IDs. As only a handful of experiences researchers can identify them while they are on sight out of the water, the Kaggle competition was created to challenge participants to automate the process using a dataset if aerial photographs of individual whales.

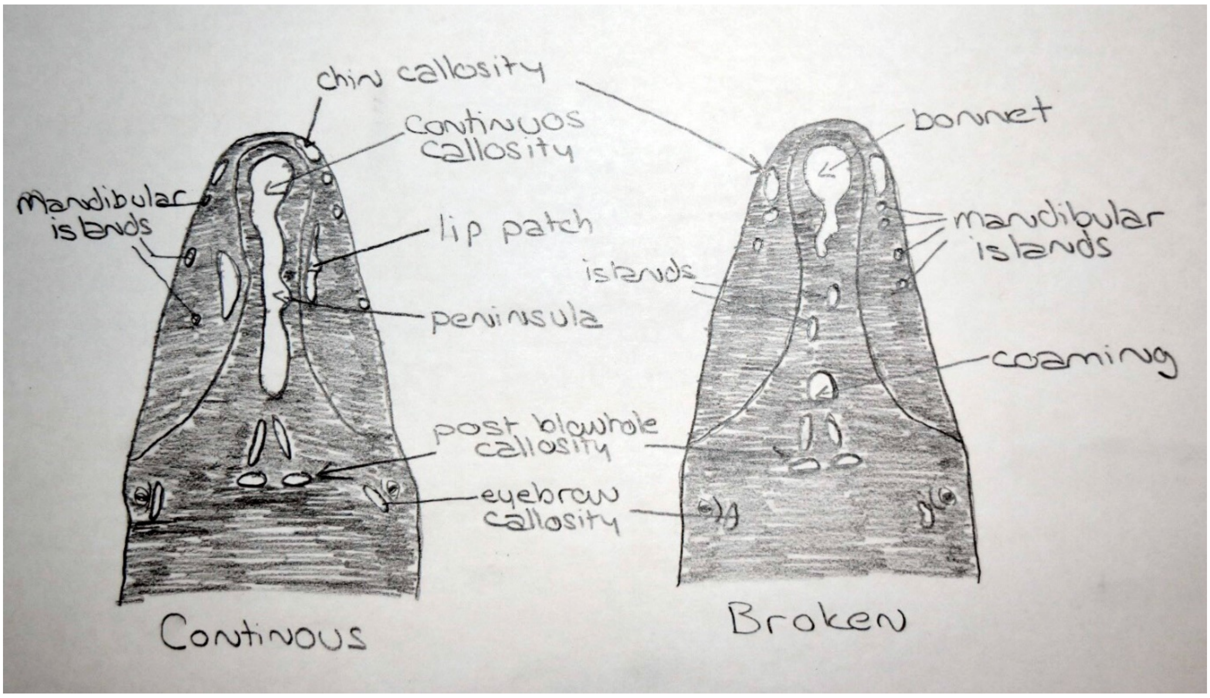
The data provided is a large set of 11,469 high resolution aerial photographs for the individual whales. A csv file containing the filenames and whale IDs for 4,544 whale images has also been provided. These 4,544 images would be our training set which will be used for training the recognition model.



Below are some sample images from the dataset.



As seen in the examples, the images contain whales in a body of water that includes splashes of water. The water body and splashes have no role in helping the model predict the whale’s ID. They rather create noise in the dataset and hinder the model from seeing important features of the whale. Upon closer inspection of the whales, a certain pattern can be noticed on their head. Christin B. Khan, a fishery biologist at NOAA explains this as the callosity pattern on the top of a whale's head. She mentions two types of callosity patterns; continuous and broken. A drawing added in discussion depicts two such patterns.



We would want our model to closely look into the head of the whale and extract features from there to learn how to classify them.

Since this is an image classification task, a convolutional neural network would be best suited to use for classifying the whales. A convolutional neural network takes in images as arrays or tensors. However, the images currently suffer from high resolution and noise which may hinder the model from focusing on the whale's head. A quick research on previously tried solutions on this problem reveals that a localisation approach should be implemented first to localise to the whale's head. Localisation is another application in computer vision and hence a neural network model can be trained to localise to the whale's head. A good localiser would return the coordinates of the image where the object to be localised exists. To accomplish this, a training dataset of images with correct coordinates are required. Researchers Bogucki et al. (2016) tried to localise using a bounding box appraoch where coordinates of the bounding boxes for whale's head in training set was added manually and a neural netowork was then trained to identify those coordinates for new images. Another approach was to identify the coordinates of the bonnethead and blowholes and use those cordinates as reference points to localise a head which was implemented by Kabani and El-Sakka (2016). The second approach seems more plausible because the neural network could learn the features of bonnethead and blowholes to correctly identify them rather than trying to find co-ordinates in the water body to get the bounding box. The approach used in this implementation is to localise the bonnethead and blowhole co-ordinates, use them to create a crop of the head as close as possible (75 pixels out from both blowhole and bonnethead) and finally use those crops to train a classifier model.

Once the localiser network is trained, the localised heads are cropped and the cropped images are used as training set to train another convolutional neural network which will be the classification model.

Data Analysis and Data Pre-processing

This section describes the steps taken to load the data, localise the head and extract head crops of the whales from the images. The file 'train.csv' contains the list of filenames and the corresponding whale IDs for whales in the training set. The first five lines loaded as a pandas dataframe are as follows.

	Image	whaleID
0	w_7812.jpg	whale_48813
1	w_4598.jpg	whale_09913
2	w_3828.jpg	whale_45062
3	w_8734.jpg	whale_74162
4	w_3251.jpg	whale_99558

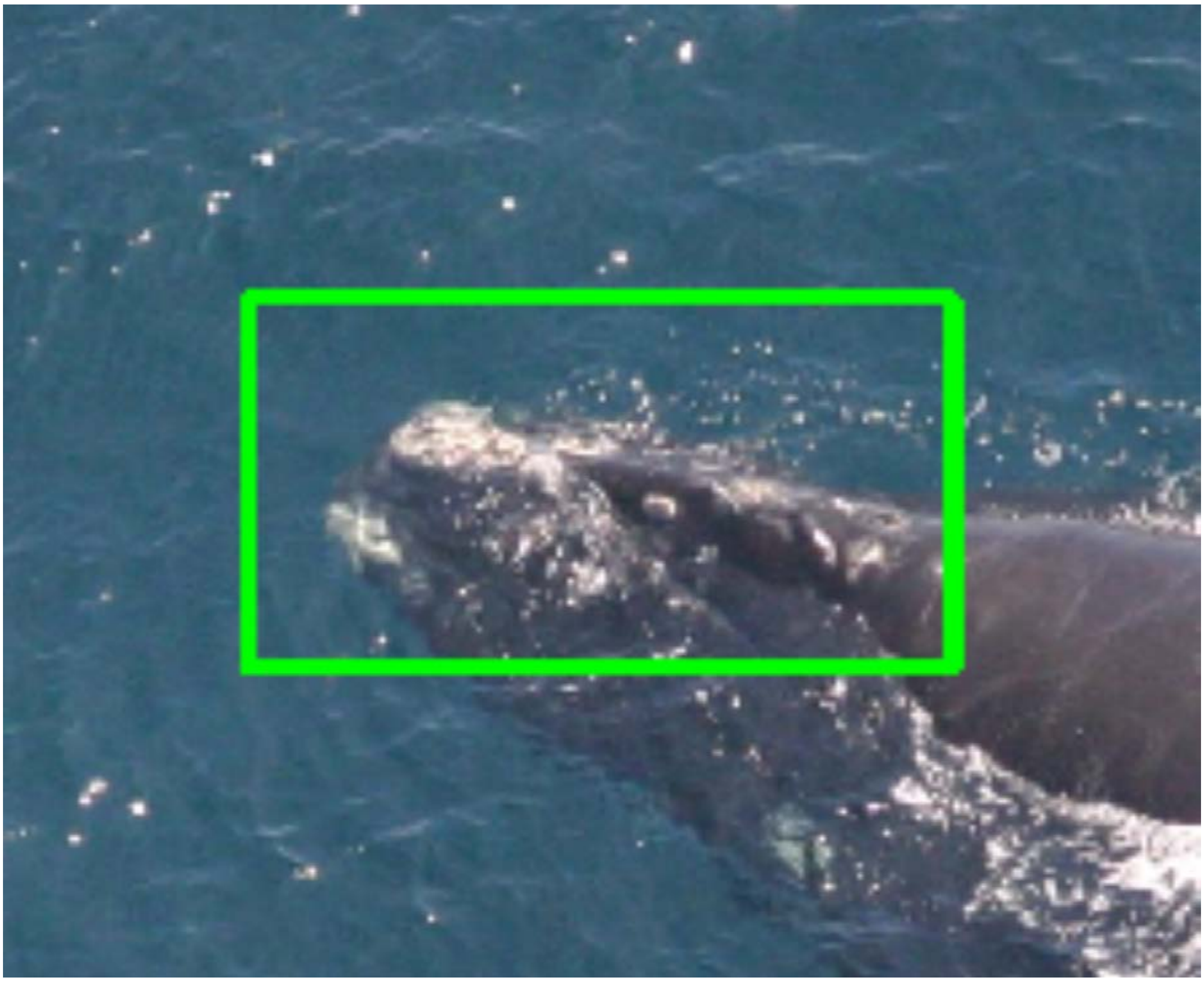
As filenames within a folder are read in alphabetical order, the train dataframe is sorted by filenames and the first five lines of resulting sorted dataframe is as follows.

	Image	whaleID
0	w_1.jpg	whale_72820
1	w_100.jpg	whale_66711
2	w_1000.jpg	whale_64496
3	w_1003.jpg	whale_48490
4	w_1004.jpg	whale_70138

As it is an image dataset, OpenCV4 and PIL image libraries were used to read the images. The images were loaded as numpy arrays. To localise the head, we needed coordinates of the whale's bonnet-head and blowholes which were generously provided in the annotations by Anil Thomas (2015) in two files named 'points1.json' and 'points2.json'. Plotting the two points in the first image of the training set returns the following annotations in the image.

This is a great reference point to have to create a head-crop. Adding 75 pixels around the coordinates gives us the following bounding box which contains within all features the classification model would need.


head localised using two points
0
250
500
750
1000
1250
1500
1750
2000
0
500
1000
1500
2000
2500
3000



With all the training set images annotated with such coordinates, a localiser network can learn to predict those coordinates for images to be classified (prediction set). A convolutional neural network was setup to learn those coordinates, of which the architecture is summarised below.

Architecture

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 128, 192, 12)      336
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 42, 64, 12)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 42, 64, 12)        1308
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 14, 21, 12)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 14, 21, 12)        1308
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 4, 7, 12)          0
_________________________________________________________________
flatten (Flatten)            (None, 336)               0
_________________________________________________________________
dense (Dense)                (None, 500)               168500
_________________________________________________________________
dense_1 (Dense)              (None, 100)               50100
_________________________________________________________________
dense_2 (Dense)              (None, 50)                5050
_________________________________________________________________
dense_3 (Dense)              (None, 10)                510
_________________________________________________________________
dense_4 (Dense)              (None, 2)                 22
=================================================================
Total params: 227,134
Trainable params: 227,134
Non-trainable params: 0
```

Since the images are of high resolution, the images were first resized to 192x128 pixels and the coordinates mapped accordingly. The localiser was then trained to predict coordinates on the resized image. The model shows convergence by the continuous reduction of loss however stalls to reduce loss after about 10 epochs. The models were saved and used to predict coordinates for the prediction set. The resulted coordinates were mapped back to the original image size and a crop from original sized image was then extracted with the mapped coordinates.

The mapped coordinates were used to create the bounding boxes for whales on prediction set and the results were analysed. While some whales were correctly localised, the model failed to localise majority of the heads. At this point a different approach to train the localiser was employed. Rather than thinking the problem as a regression problem to approximate pixel coordinates, the problem was turned into a multiclass classification problem with all possible pixels as classes. This gave better predictions for the bonnet-head and blowhole coordinates resulting into better head-crops being extracted.

The extracted head-crops were loaded into numpy array ready for the classification model.

Legal, Ethical and Privacy Concerns

The data for this challenge was collected by National Oceanic and Atmospheric Administration (NOAA) Fisheries. The data was collected over a course of 10 years and hundreds of helicopter trips paid for by the NOAA. As such, the copyright of the data belongs to NOAA and a researcher must only use the data for the challenge provided. For any other usage, the researchers should seek consent from the NOAA. This requirement is also within the regulations maintained by the General Data Protection Regulation (GDPR) which currently applies in the United Kingdom. (Gruschka et al., 2018) Moreover, the images provided on the dataset must not be shared online without citing the source to avoid copyright infringement. As the images do not contain any sensitive human information, most of the data protection regulations do not apply. However, the images and the whale IDs are sensitive dataset in respect of wildlife conservation. Furthermore, since the whales pictured in the dataset belong to an endangered species, it is important to ensure that the images and the whale IDs do not reach the hands of potential poachers.

Classification

The classification model had the cropped training images as inputs and their corresponding one-hot encoded whale IDs as targets. After creating a train test split, a convolutional neural network was trained to classify the images. Below is a summary of the network that was used.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 128, 256, 24)      672
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 42, 85, 24)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 42, 85, 24)        5208
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 14, 28, 24)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 14, 28, 24)        5208
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 4, 9, 24)          0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 4, 9, 24)          5208
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 1, 3, 24)          0
_________________________________________________________________
flatten (Flatten)            (None, 72)                0
_________________________________________________________________
dense (Dense)                (None, 1000)              73000
_________________________________________________________________
dense_1 (Dense)              (None, 447)               447447
=================================================================
Total params: 536,743
Trainable params: 536,743
Non-trainable params: 0
```

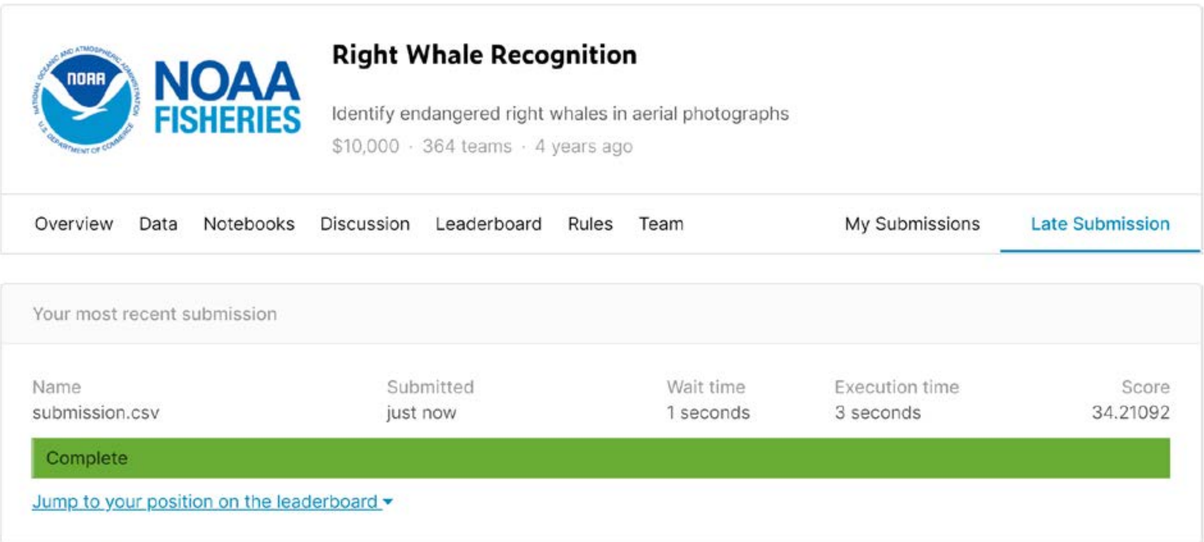
The model 3 convolutional layers each followed by max pooling layers. The convolutional layers had (3,3) kernels and same padding to keep the size of the feature maps same as inout as it passes through layers. The model was optimised using categorical crossentropy loss and softmax activation for output dense layer for classification.

VGG-16 and Inception-V4 were the pre-trained networks which were downloaded with their imagenet weights and trained for classiffying the cropped images. However, both resulted in poor training performance and negligible change in validation loss.

Finally, a model with the latest squeeze-and-excitation block as proposed by Hu et al. (2019) was created for which the network is summarised as below. According to Hu et al. (2019), the squeeze and excitation block adaptively recalibrates channel-wise feature responses by explicitly modelling interdependencies between channels. Although this network did not result in significant improvement, it did show a slightly better validation accuracy and submission loss.

```
Model: "model_6"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_7 (InputLayer)            [(None, 128, 256, 3) 0
__________________________________________________________________________________________________
conv2d_24 (Conv2D)              (None, 128, 256, 24) 672         input_7[0][0]
__________________________________________________________________________________________________
conv2d_25 (Conv2D)              (None, 128, 256, 24) 5208        conv2d_24[0][0]
__________________________________________________________________________________________________
conv2d_26 (Conv2D)              (None, 128, 256, 24) 5208        conv2d_25[0][0]
__________________________________________________________________________________________________
global_average_pooling2d_14 (Gl (None, 24)           0           conv2d_26[0][0]
__________________________________________________________________________________________________
reshape_8 (Reshape)             (None, 1, 24)        0           global_average_pooling2d_14[0][0]
__________________________________________________________________________________________________
dense_22 (Dense)                (None, 1, 0)         0           reshape_8[0][0]
__________________________________________________________________________________________________
dense_23 (Dense)                (None, 1, 24)        24          dense_22[0][0]
__________________________________________________________________________________________________
multiply_8 (Multiply)           (None, 128, 256, 24) 0           conv2d_26[0][0]
                                                                 dense_23[0][0]
__________________________________________________________________________________________________
average_pooling2d_7 (AveragePoo (None, 64, 128, 24)  0           multiply_8[0][0]
__________________________________________________________________________________________________
global_max_pooling2d_6 (GlobalM (None, 24)           0           average_pooling2d_7[0][0]
__________________________________________________________________________________________________
global_average_pooling2d_15 (Gl (None, 24)           0           average_pooling2d_7[0][0]
__________________________________________________________________________________________________
concatenate_6 (Concatenate)     (None, 48)           0           global_max_pooling2d_6[0][0]
                                                                 global_average_pooling2d_15[0][0]
__________________________________________________________________________________________________
dense_24 (Dense)                (None, 447)          21456       concatenate_6[0][0]
==================================================================================================
Total params: 32,568
Trainable params: 32,568
Non-trainable params: 0
```

Kaggle Submission Proof



Result and Scope for future Improvement

The results of the challenge in this case is not so satisfactory because the loss score observed is only slightly better than the sample submission benchmark. The first reason for this could be the overfitting that occurs within the model. The images that were cropped still had water areas within them and the head was aligned in different directions. A head aligning mechanism could be introduced during cropping so that the heads are always facing in the same direction. Secondly, the cropping could be done for only the callosity pattern area within the head.

Other methods that could be employed are image manipulation to brighten the callosity pattern within the head.

Similarly, the localisation networks had an accuracy of about 90% only. Since, it was a regression task to predict co-ordinates, this accuracy may not be the best and can result in several images not being cropped properly.

Below we see compare of the cropped images from training set and prediction set.

Training Set:

Prediction Set:

As see, the training images were spot on with their crops but prediction images suffered from some crops not having the whale at all. This occurred because the accuracy during training was around 90% and hence not a perfect predcition of bonnethead and blowhole co-ordinates. Manual annotation of bonnethead and blowhole in the prediction will help resolve this. Moreover, dividing the image areas into bins and doing a classification task to predict head area could be another option to try.

Finally, to check if the validation accuracy increases, a closer crop of the training images was extracted as shown below. However, the model still failed to converge. Researchers Bogucki et al. (2016) suggest aligning the head in only one direction to make the training model’s life easier for prediction.

Conclusion

This was an image recognition task where North Atlantic Right whales were to be recognized with their IDs by scanning their callosity pattern in the head. The task had two challenges, first was the localisation and second was the recognition itself. Two localisers trained to find the bonnethead and blowhole co-ordinates were trained. Although with about 90% accuracy, the models could not pinpoint the bonnethead and blowhole accuracy for many whales. Similarly, the recognition model that was trained on cropped training images failed to generalise for the validation set.

References

Bogucki, R., Cygan, M., Klimek, M., Milczek, J. K. and Mucha, M., 2016. *Which whale is it, anyway? Face recognition for right whales using deep learning*, [online] Available at: <https://deepsense.ai/deep-learning-right-whale-recognition-kaggle/> [Accessed 16/04/2020]

Gruschka, N., Mavroeidis, V., Vishi, K. and Jensen, M., 2018. Privacy Issues and Data Protection in Big Data: A Case Study Analysis under GDPR, *ArXiv*, [PDF]. Available at: <https://arxiv.org/pdf/1811.08531.pdf> [Accessed on: 11 December 2019]

Hu, J., Shen, Li., Albanie, S., Sun, G and Wu, E., 2019. Squeeze-and-Excitation Networks, ArXiv, [online] Available at: <https://arxiv.org/pdf/1709.01507.pdf> [Accessed 23/04/2020]

Kabani, A. and El-Sakka, M. R., 2016. North Atlantic Right Whale Localization and Recognition using very Deep and Leaky Neural Network, *Mathematics for Applications*, 5, pp.155-170

Khan, C., 2015. *Right Whale Questions*. [online[] Available at: <https://www.kaggle.com/c/noaa-right-whale-recognition/discussion/16172> [Accessed 14/04/2020]

Thomas, A., 2015. *Right Whale Recognition*, [online] Available at: <https://github.com/anlthms/whale-2015> [Accessed 17/04/2020]