UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

CORSO DI LAUREA IN INGEGNERIA INFORMATICA

"HIERARCHICAL NEURAL-NETWORK ENSEMBLES FOR SHARK LOCATION AND CLASSIFICATION"

Relatore: Prof. Loris Nanni

Correlatori: Prof. Francesco Ferretti

Dott. Jeremy Jenriette

Laureando: Elia Feltrin

ANNO ACCADEMICO 2021 - 2022

Index

Abstract

English version

Sharks are excellent indicators of ocean environmental health and suitable shark's conservation depends on well-informed population assessments. Scientific surveys and fisheries monitoring are adequate to define population status, but species-specific indices of abundance and distribution, coming from this source, are rare for most shark species. That is the reason why boost media-based remote monitoring with machine learning helps scientists of several areas. Previous work (Jenrette et al. *Shark detection and classification with Machine Learning)* based on hierarchical locators and classifiers reaches about 70% species-classification accuracy. We managed to improve this method by replacing every classifier with an ensemble composed of CNNs that have been trained with several activation functions.

Versione italiana

Gli squali sono eccellenti indicatori della salute degli oceani e un'appropriata conservazione di questa specie dipende da una ben informata valutazione della popolazione. Sondaggi scientifici e controllo della pesca risultano essere metodi adeguati per definire lo stato della popolazione, ma indici che misurano la ricchezza e la distribuzione degli squali negli oceani sono difficilmente ottenibili da queste fonti per molte specie. Per questo motivo incrementare il monitoraggio con tecniche di machine learning risulta utile per gli studiosi di diverse aree scientifiche. In un lavoro precedente (Jenrette et al. Shark detection and classification with Machine Learning) si è riusciti ad ottenere accuracy intorno al 70% nella classificazione delle specie. Questo metodo è stato migliorato, in collaborazione con I ricercatori che hanno sviluppato il progetto sopracitato, sostituendo le reti neurali di classificazione e localizzazione con ensembles di CNNs rese parzialmente indipendenti attraverso la modifica delle funzioni di attivazione.

INTRODUCTION:

Shark's health is constantly challenged by growing fishing pressure, as well as poor management and conservation as a result of data paucity, insufficient taxonomic knowledge and underdeveloped monitoring methods [1]. Due to the high cost of shark monitoring via surveying and fisheries, exacerbated by large home ranges of some species [2], sharks remain an extremely data deficient group of marine animals [3] [1]. A recent study [4] produced a hierarchical CNN-based detector called "Shark Detector". It can generate, detect and classify shark-sourced visual media and efficiently post-process Baited Remote Underwater Videos (BRUVs), camera trap images, Remotely Operated underwater Vehicle (ROV) footage and shared social media by removing irrelevant content and classifying shark species. However, shark's classification is not a straightforward process for machine learning since many morphologically diverse and data-poor shark species can be found in nature. For this reason the purpose of this work is to improve the performance of the Shark Detector.

In order to obtain more reliable results, we replaced single models with network ensembles: based on [5], a good way to obtain partially independent classifiers is to replace the CNNs' original activation function layers with different ones. The hierarchical structure of the entire Shark Detector was re-implemented according to the high-performance ensemble method proposed by Nanni et al. [5].

1. ESSENTIAL KNOWLEDGE FROM RELATED WORK

1.1 Shark Detector: original structure

The original pipeline of the Shark Detector (referred as SD from now on) is composed by three main components:

1. Shark Locator (SL): an object detection model that locates shark subjects in images and draws bounding boxes around them;
2. Shark Identifier (SI): a stronger binary classifier which can determine with greater accuracy which images depict sharks and which not;
3. Shark Classifiers (SCs): multiclass models that classify shark images by genus and species.

The shark training images used to develop these models were mainly sourced from sharkPulse, a crowd-sourcing platform which mines and aggregates shark's media from social networks, citizen science and projects, users' submission and other electronic archives [3].

Shark Locator

This model locates and extracts shark subjects by cropping one or multiple of them from pictures, thus creating new images for the dataset. Thank to this processes, two goals are achieved: noisy backgrounds, which challenge the training process, are removed and multiple-subject images are split in several single-subject images. In this way, during the SI training process, the shark dataset can be boosted (from 24,546 to 53,345 images), while during media detection all subjects can be classified individually. To build the SL, Jenriette and his team sourced Tensorflow's Model Garden [6] and used a Faster Region-based Convolutional Neural Network (Faster-RCNN) algorithm. The model was trained with the Common Objects in Context (COCO) dataset (consisting of 236 shark images) to detect and draw boxes around sharks [7][8].

Shark Identifier

This step consists in a binary sorting model. The SI filters out non-shark images before the remaining ones are taxonomically classified by the following models. Jenriette and his team sourced 53,345 shark images from Instagram and sharkPulse and 50,260 non-shark pictures just from Instagram. The original Shark Identifier learns key shark features from training images by optimizing the binary cross-entropy loss function [9]. In order to reduce the number

of required training steps, a pre-trained model was incorporated (this is known as transfer learning). The SI was pre-trained with the VGG16 network, which is trained on 1.28 million images with 1000 categories.

To increase accuracy, image augmentation techniques like shifting, shearing, zooming and rotation were applied [10][11]. The pre-trained network was modified in order to perform fine-tuning and the output neurons (shark and non-shark) were normalized with the SoftMax activation function [12]. The model was trained with 90% of the training dataset and validated on the remaining 10% over 10 epochs.

Shark Classifier

The last component of the Shark Detector is a hierarchical classification framework to classify shark images taxonomically. It is composed by a genus-specific and several species-specific models, one for each genus. Filtered shark images are ingested by the genus-specific classifier (GSC), that redirects them to the correct species-specific classifier (SSCg), which then labels them with the most likely species.

The Shark Classifier was trained with the sharkPulse database, images cropped with the SL and Instagram images. Thus, the database consists in 84 genera and 219 species and contains 36,722 genus-labeled images and 19,243 species-labeled images with an average of 167 images per species. It was estimated that an average of 433 ± 47 images is needed to produce at least 50% recall for the GSC and in the same way the tests show that an average of 161 ± 41 images is necessary to produce at least 50% recall for the SSCs. Jenriette and his team used these averages as training data quantity thresholds for the SC. The models were optimized with the categorical cross-entropy function to learn shark features and DenseNet201 was incorporated as a pre-trained network for multi-class classification. Tests show that the sigmoid activation function leads to a higher test accuracy compared to ReLU.

In conclusion, the SD is able to detect 26 genus and 47 species that meet the training data thresholds.

For more details on SD implementations refer to [4].

1.2 Stochastic activation function

Activation functions (AFs) play a crucial role in discriminative capabilities of CNNs and the design of such new functions is an active area of research. Activation functions can be divided into two types: static or dynamic. The first ones consider all neurons and layers as identical, the second learn their own parameters for each layer or even each neuron independently. Although dynamic activation functions perform better in some applications, an increasing number of trainable parameters can lead to overfitting. [5]

A recent study [5] proposed a new method to obtain a high-performing classifier ensemble: every AF of each model of the ensemble is chosen randomly from a set that includes several different AFs. This method was called “Stochastic Selection of Activation Layer for CNNs” (SSAL). For more details regarding the AFs included and their definition see [5]. SSAL method reaches good results:

1. the SSAL ensembles always perform better than the stand-alone methods, so SSAL is a good way to build partially independent models;

2. the ReLu AF, that is canon when utilizing ResNet50, is not the best. However, there is not an AF that performs always better than the others, therefore an ensemble is the only way to ensure that the best results are included.

3. single models designed by meaning of SSAL perform better than base-line models and a little worse than ensembles.

4. small SSAL ensembles (three models) strongly outperform stand-alone approaches and reach performances comparable to “heavier” ensembles (10 or 20 models).

1.3 YOLO v4

YOLO, or You Only Look Once, is a fast object detection family of algorithms that is often used in real-time applications. The basic idea proposed by Joseph Redmon [15] is the first of many iterations this architecture has gone through: YOLO combines what was once a multi-step process in a single neural network that performs both classification and prediction of bounding boxes for detected objects. One of the newer versions of this architecture, called YOLO version4 (YOLOv4, hereinafter referred just as YOLO), is mainly composed of three parts:

1. backbone: it is the deep learning architecture that acts as a feature extractor. It is usually a classification model, for example ResNet50;
2. neck: it basically collects feature maps from different stages of the backbones;
3. head: it finds the region where the object might be present, but it does not give any information about the object present.

2. PROPOSED METHOD

This work aims to apply the SSAL method to the Shark Detector. To achieve this each component of the SD (SL, SI, GSC and SSCs), that originally was a single classifier, becomes an ensemble of classifiers. These new components are called Shark Locator Ensemble (SLE), Shark Identifier Ensemble (SIE), Genus Shark Classifier Ensemble (GSCE) and Species Shark Classifier Ensembles (SSCEs). The entire detection framework, consisting of SLE, SIE, GSCE and SSCEs is referred to as Shark Detector Ensembles (SDEs).

2.1 SIE, GSCE and SSCEs

The development of the classifier ensembles is similar to each classification step of the SDEs and is mainly composed of three steps:

1. **Create partially independent classifiers**
   For each classification component of the SDEs (SIE, GSCE, SSCEs) we applied the SSAL method on three ensembles, each composed by:
   - 15 Resnet50 [16];
   - 5 MobileNetv2 [17];
   - 5 GoogleNet [18].

   These networks were pre-trained on the imageNet dataset [13]. We chose to train more ResNet50-based models because, based on Nanni et al. unpublished data, ResNet50 is more sensitive than other models regarding AF changing. However, in order to obtain networks that are as independent as possible, we included some other models.

2. **Train networks**
   All models' input layer size is 224x224 pixels, on three color levels (Red, Green, Blue). We resized all the images to fit the models' input layer. SIE, GSCE and SSCE models were trained over 20 epochs. All models were trained with SGDM [14] optimizer and we set the initial learning rate to $3x10^{-4}$.

   The last three fully-connected layers were modified in order to have the correct number of output neurons which were normalized with the SoftMax [12] activation function, therefore their output could be interpreted as a probability.

The datasets composition is reported in tables 5.1, 5.2 and 5.3.

3. **Evaluate ensembles**

   We created several ensembles that include a variable number of models, for a total of 25 ensembles. We evaluated all the ensembles to understand which are the best ones and which are the most recurrent models among those. In order to better evaluate the performance of the ensembles, we calculated three metrics: accuracy, precision and recall. The first one measures how many patterns are correctly predicted by the classifier over the total number of patterns; second one represents what proportion of positive identifications are actually correct while last one shows what proportion of actual positive are identified correctly.

2.2 SLE

As for the SLE development, we proceeded as following:

1. **Create partially independent locators**

   We used each one of the trained network of SIE as backbone for YOLOv4, thus creating an ensemble of locator.

2. **Train networks**

   We trained these models on 700 shark images, each containing a variable number of bounding boxes, from one to seven.

3. **Create all possible ensembles of locators:**

   As it was done for SIE, SGCE and SSCEs, we recombined the trained networks in several ensembles.

4. **Merge bounding boxes from ensembles of locators**

   Unlike the classification results, merging bounding boxes from several locators are not trivial. Main problem is that there is not a priori knowledge of which bounding boxes, created by different locators, are related to the same subject in the image. To solve this problem, given N total bounding boxes as the output of the SLE, we iterated a clustering algorithm N times, setting at each iteration the number of output clusters equal to N. For each iteration we calculated the sum of the distances between all bounding boxes and all clusters. The algorithm calculates the distance between two patterns as a function of the Euclidian distance between the centers and the intersection over union (IoU) of the two bounding boxes. This value decreases with the

increase of the number of output clusters, but it presents a significant step in correspondence of the effective number of subjects in the image. After calculating the number S of subjects in the image and the relative clusters C_i, the algorithm merges all bounding boxes of C_i in S bounding boxes, computed as the mean of all bounding boxes in C_i.

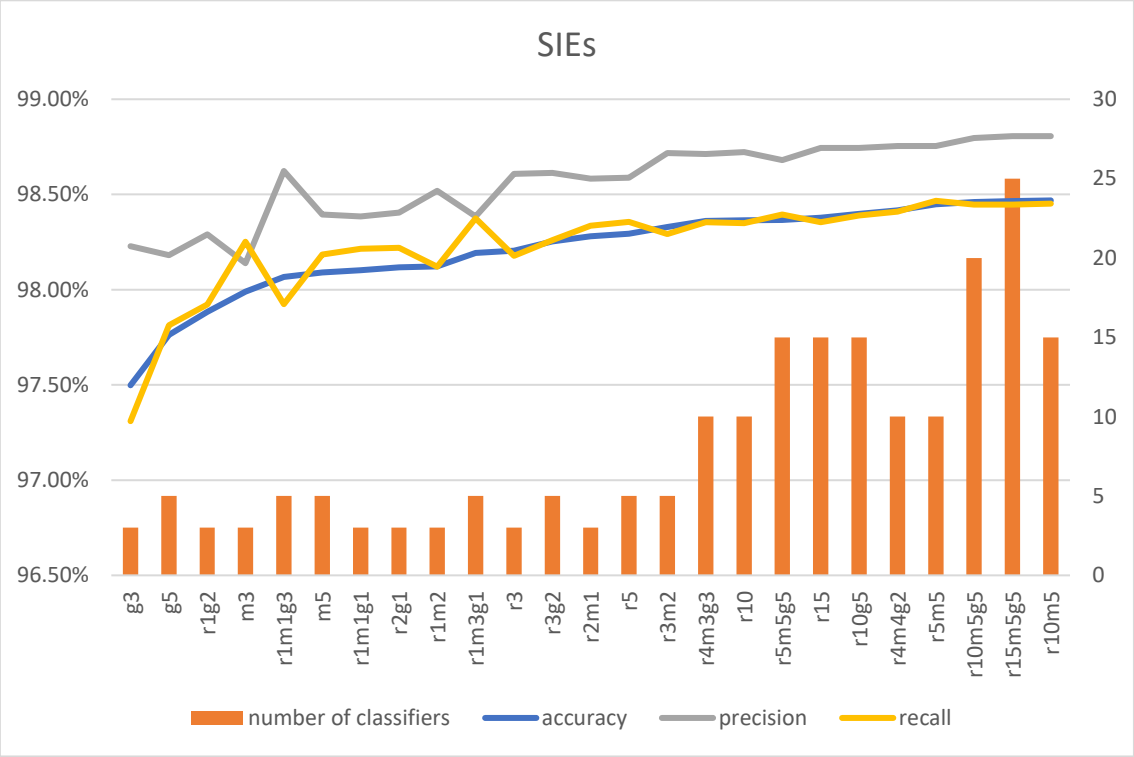
5. **Evaluate ensembles**

   At this stage we evaluated how many subjects had been found by the locator and the IoU between the ground-truth bounding boxes and the predicted ones.

3. APPLICATION AND RESULTS

3.1 SIE

It is not trivial that ensembles with several classifiers perform better than smaller ones. That is why we tested several ensembles that differ in size and composition. We evaluated the accuracy for each ensemble by putting it in relation to how many and which classifiers are more present.



On the horizontal axes the names of ensembles are shown. The letters represent the following models:

- “r”: SSAL-modified ResNet50;
- “g”: SSAL-modified GoogleNet;
- “m”: SSAL-modified MobileNetv2;

and the numbers how many of those models are included.

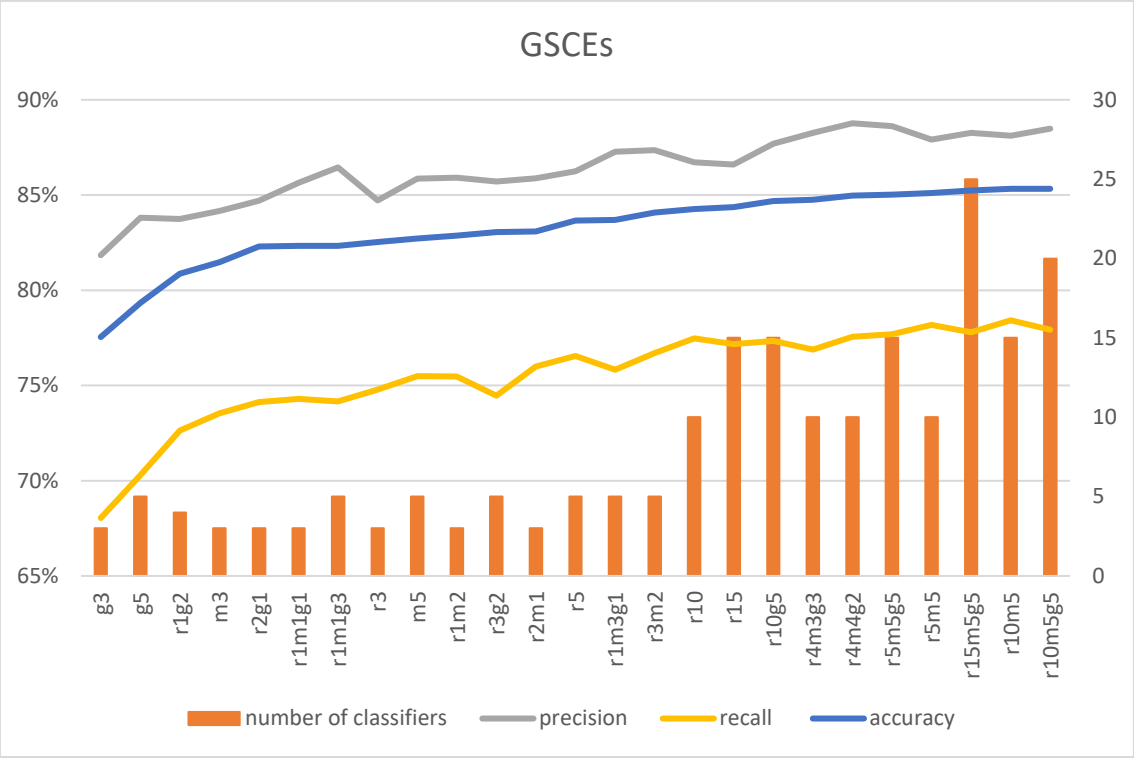
All ensembles were sorted by the accuracy value. Accuracy, recall and precision values of all ensembles are represented by the blue, yellow and grey line respectively, while the orange bars show the number of classifiers in correspondence of the metrics values. We can observe that:

- In most cases, bigger ensembles perform better than smaller ones;
- The biggest ensemble is not the best one;
- Ensembles composed by 10 classifiers reach accuracy values comparable with the best ensemble;
- Among the ensembles containing the same number of classifiers, the better ranked are the ones containing more SSAL-modified ResNet50 in combination with SSAL-modified MobileNet.

For more detailed results see table 5.4 and 5.5.

3.2 GSCE

As shown in table [5.2], data quantity is badly distributed among classes. An unbalanced training set could lead to inaccurate classifier: the classifier could be pushed to ignore the less represented classes, because the classification error on those weights significantly less than the one on most represented classes. In order to determine if this problem is present and, if needed, pinpoint the minimum quantity of patterns that are necessary to obtain a well-trained network, we trained a smaller ensemble consisting of five SSAL-modified ResNet50 and five SSAL-modified MobileNetv2 on the GSCE dataset over all its 85 classes. Then, we tested all possible ensembles and we discovered that the one formed by all the trained classifiers performs better than all the others and it reaches an accuracy of to 84.73%. We calculated the accuracy for each class individually and we find out that the following eight genera present an accuracy value lower or equal to 50%: *Holohalaelurus, Mollisquama, Odontaspis, Parmaturus, Pristis, Proscyllium, Pseudocarcharias, Pseudotriakis.* Since these genera cannot be treated individually, we grouped them together in a class called *other genera.* After that, we trained the GSCE on the datasets with merged genera. The following chart represents the performance of the GSCEs. It can be observed from that the general trend is similar to the SIEs one and the considerations aforementioned apply.



3.3 SSCEs

As shown in table 5.3, there are many genera that contain only one labeled species: in these cases, there is not enough data to train a local SSCE and the classification stops at genus level. For the remaining 41 genera, the problem of unbalanced species data quantity also recurred, so we trained a smaller ensemble of classifiers consisting of five SSAL-ResNet50 and five SSAL-MobileNetv2.

As for GSCE, we evaluated which species are well represented in the SSCE dataset by using the small ensemble: the SSCEs that did not reach an overall accuracy of 50% or with a single-species accuracy lower or equal to 50% have not been included in the final SSCEs training.

For the remaining classes, SSCEs was trained.

At time of writing, we cannot provide the training results of neither SSCEs nor SLE due the large amount of time that these trainings require. At the moment, we are training the smallest ensemble in order to understand which SSCEs should be trained. After that, we will proceed to train SLE and select the best ensemble in order to assemble the entire SDE pipeline.

4. CONCLUSIONS

The proposed method was found to be effective in improving the performance of both SI and GSC. The binary classification task can be now performed with an accuracy of 98.47%, while the original SI reaches approximately 91% accuracy. The genus classification task can be performed with an accuracy of 85.33%, outperforming GSC by about 15%. The performance increase due to the use of ensembles is remarkable especially for GSC: its networks reach an accuracy value of 81.06% while its best ensemble gets up to 85.33%; SIE presents a performance increase lower than the GSCE: the single networks of SIE are able to classify with an accuracy of 98% while its best ensemble, that includes 15 models, increased this value of 0.47%. SSCEs and SLE are still under development due the high computing time that is required to train such complex models. However, the excellent results obtained until now give us a reason to be optimistic. More tests and results will follow.

5. APPENDICIES

5.1 SIE dataset

Classes	Patterns
Shark	64449
Not Shark	50260
Total	**114709**

5.2 GSCE dataset

Class	Patterns	Class	Patterns
Aculeola	32	Hypogaleus	16
Alopias	1226	Isistius	80
Anoxypristis	27	Isogomphodon	28
Apristurus	244	Isurus	1635
Asymbolus	420	Lamna	405
Atelomycterus	102	Loxodon	115
Brachaelurus	479	Megachasma	349
Callorhinchus	145	Mitsukurina	51
Carcharhinus	5700	Mollisquama	5
Carcharias	2404	Mustelus	677
Carcharodon	2245	Nebrius	343
Centrophorus	178	Negaprion	905
Centroscyllium	44	Notorynchus	307
Centroscymnus	90	Odontaspis	66
Cephaloscyllium	861	Orectolobus	2021
Cetorhinus	638	Oxynotus	109
Chaenogaleus	149	Parascyllium	100
Chiloscyllium	326	Parmaturus	12
Chimaera	241	Poroderma	404
Chlamydoselachus	116	Prionace	985
Cirrhigaleus	90	Pristiophorus	19
Dalatias	187	Pristis	84
Deania	118	Proscyllium	23
Echinorhinus	516d	Pseudocarcharias	33
Etmopterus	233	Pseudoginglymostoma	37
Eucrossorhinus	404	Pseudotriakis	15
Eusphyra	76	Rhincodon	1649
Galeocerdo	1115	Rhizoprionodon	398
Galeorhinus	791	Schroederichthys	147
Galeus	575	Scoliodon	73
Ginglymostoma	1142	Scyliorhinus	964
Glyphis	82	Scymnodon	21
Gollum	18	Somniosus	103
Halaelurus	41	Sphyrna	1591
Haploblepharus	691	Squaliolus	23
Hemigaleus	54	Squalus	1044
Hemipristis	149	Squatina	316
Hemiscyllium	308	Stegostoma	343
Heptranchias	77	Sutorectus	50
Heterodontus	2244	Triaenodon	1857
Hexanchus	987	Triakis	1146
Holohalaelurus	15	Trigonognathus	11
Hydrolagus	436		
Total:		**44576**	

5.3 SSCEs dataset

Genus	Number of classes	Total patterns
Aculeola	1	4
Alopias	3	527
Apristurus	6	41
Asymbolus	4	30
Atelomycterus	2	33
Brachaelurus	2	461
Callorhinchus	3	3
Carcharhinus	24	3800
Carcharias	1	2405
Carcharodon	1	2290
Centrophorus	5	30
Centroscyllium	2	8
Centroscymnus	4	11
Cephaloscyllium	4	663
Cetorhinus	1	642
Chaenogaleus	1	1
Chiloscyllium	5	299
Chimaera	1	4
Chlamydoselachus	1	6
Cirrhigaleus	2	5
Dalatias	1	44
Deania	2	12
Echinorhinus	2	458
Etmopterus	7	25
Eucrossorhinus	1	416
Eusphyra	1	7
Galeocerdo	1	1117
Galeorhinus	1	791
Galeus	1	376
Ginglymostoma	2	945
Glyphis	2	9
Gollum	1	3
Haploblepharus	4	680
Hemigaleus	2	6
Hemipristis	1	13
Hemiscyllium	8	217
Heptranchias	1	18
Heterodontus	7	2180
Hexanchus	3	800
Hydrolagus	3	391
Hypogaleus	1	2
Isistius	1	30
Isurus	2	1422
Lamna	2	241
Loxodon	1	18
Megachasma	1	349
Mitsukurina	1	8
Mollisquama	1	1
Mustelus	12	522
Nebrius	1	343
Negaprion	2	240
Notorynchus	1	307
Odontaspis	1	38
Orectolobus	9	1939
Oxynotus	2	43
Parascyllium	3	33
Poroderma	2	374

Prionace	1	990
Pristiophorus	2	8
Pseudocarcharias	1	1
Pseudoginglymostoma	1	3
Rhincodon	1	1602
Rhizoprionodon	7	344
Schroederichthys	2	38
Scoliodon	1	3
Scyliorhinus	3	472
Somniosus	3	30
Sphyrna	7	954
Squaliolus	1	1
Squalus	10	312
Squatina	12	298
Stegostoma	1	277
Sutorectus	1	26
Triaenodon	1	1786
Triakis	4	1059
Total	**226**	**33885**

5.4 SIE single networks results

model	accuracy	precision	recall
SSAL-ResNet50	97.71%	97.90%	97.99%
	97.83%	98.29%	97.83%
	97.93%	**98.49%**	97.81%
	97.92%	98.03%	98.24%
	97.74%	97.86%	98.08%
	98.06%	98.48%	98.05%
	97.95%	98.40%	97.94%
	97.81%	98.04%	98.04%
	97.93%	98.20%	98.08%
	97.92%	98.35%	97.92%
	97.95%	98.37%	97.97%
	97.84%	98.25%	97.89%
	98.02%	98.26%	98.19%
	97.93%	98.27%	98.02%
	97.90%	98.23%	98.00%
SSAL-MobileNetv2	97.61%	97.98%	97.74%
	97.20%	97.79%	97.21%
	97.20%	96.69%	**98.26%**
	97.32%	98.02%	97.20%
	97.57%	98.01%	97.64%
SSAL-GoogleNet	96.80%	97.00%	97.26%
	97.27%	98.05%	97.09%
	96.84%	98.27%	96.15%
	97.05%	97.00%	97.70%
	97.44%	97.35%	98.05%

5.5 SIEs results

name	accuracy	precision	recall	SSAL-Resnet50	SSAL-MobileNet	SSAL-GoogleNet	Total classifier
r3	98,20%	98,61%	98,18%	3	0	0	3
m3	97,99%	98,14%	98,25%	0	3	0	3
g3	97,50%	98,23%	97,31%	0	0	3	3
r2m1	98,28%	98,58%	98,34%	2	1	0	3
r2g1	98,12%	98,40%	98,22%	2	0	1	3
r1m2	98,12%	98,52%	98,12%	1	2	0	3
r1g2	97,88%	98,29%	97,92%	1	0	2	3
r1m1g1	98,10%	98,38%	98,22%	1	1	1	3
r5	98,29%	98,59%	98,36%	5	0	0	5
m5	98,09%	98,39%	98,18%	0	5	0	5
g5	97,76%	98,18%	97,81%	0	0	5	5
r3m2	98,33%	98,72%	98,29%	3	2	0	5
r3g2	98,25%	98,61%	98,26%	3	0	2	5
r1m3g1	98,19%	98,38%	98,37%	1	3	1	5
r1m1g3	98,07%	98,62%	97,92%	1	1	3	5
r10	98,36%	98,72%	98,35%	10	0	0	10
r5m5	98,45%	98,75%	**98,47%**	5	5	0	10
r4m3g3	98,36%	98,71%	98,35%	4	3	3	10
r4m4g2	98,42%	98,75%	98,41%	4	4	2	10
r15	98,38%	98,74%	98,35%	15	0	0	15
r10m5	**98,47%**	**98,81%**	98,45%	10	5	0	15
r10g5	98,40%	98,74%	98,39%	10	0	5	15
r5m5g5	98,37%	98,68%	98,39%	5	5	5	15
r10m5g5	98,46%	98,80%	98,45%	10	5	5	20
r15m5g5	98,47%	98,81%	98,45%	15	5	5	25

5.6 GSCE single networks results

model	precision	recall	accuracy
SSAL-ResNet50	79.21%	73.17%	79.65%
	79.44%	71.83%	79.63%
	80.03%	72.51%	80.17%
	79.20%	73.55%	80.10%
	81.15%	74.04%	80.67%
	80.16%	72.90%	79.28%
	80.62%	73.36%	79.82%
	82.35%	**74.54%**	80.62%
	80.71%	73.35%	79.95%
	80.56%	72.55%	80.11%
	80.39%	73.97%	80.38%
	79.82%	73.81%	**81.06%**
	79.70%	70.18%	78.97%
	80.40%	71.77%	79.78%
	81.61%	73.93%	80.79%
SSAL-MobileNetv2	77.96%	69.54%	77.16%
	78.38%	69.58%	77.05%
	78.46%	69.14%	76.71%
	78.21%	70.68%	77.86%
	77.10%	70.54%	77.09%
SSAL-GoogleNet	73.12%	60.77%	71.19%
	71.52%	65.26%	73.42%
	74.22%	62.09%	71.97%
	74.11%	65.93%	73.07%
	75.09%	61.56%	72.01%

5.7 GSCEs results

ensemble	accuracy	precision	recall	SSAL-ResNet50	SSAL-MobileNet	SSAL-GoogleNet	Total classifiers
r3	82.54%	84.72%	74.78%	3	0	0	3
m3	81.47%	84.16%	73.55%	0	3	0	3
g3	77.54%	81.84%	68.04%	0	0	3	3
r2m1	83.09%	85.88%	75.99%	2	1	0	3
r2g1	82.30%	84.70%	74.12%	2	0	1	3
r1m2	82.88%	85.91%	75.47%	1	2	0	3
r1g2	80.86%	83.74%	72.63%	1	0	2	3
r1m1g1	82.33%	85.64%	74.30%	1	1	1	3
r5	83.66%	86.25%	76.54%	5	0	0	5
m5	82.71%	85.86%	75.49%	0	5	0	5
g5	79.34%	83.82%	70.29%	0	0	5	5
r3m2	84.07%	87.36%	76.71%	3	2	0	5
r3g2	83.06%	85.71%	74.47%	3	0	2	5
r1m3g1	83.70%	87.28%	75.83%	1	3	1	5
r1m1g3	82.34%	86.46%	74.16%	1	1	3	5
r10	84.27%	86.73%	77.47%	10	0	0	10
r5m5	85.10%	87.91%	78.17%	5	5	0	10
r4m3g3	84.75%	88.26%	76.88%	4	3	3	10
r4m4g2	84.98%	88.77%	77.56%	4	4	2	10
r15	84.36%	86.60%	77.17%	15	0	0	15
r10m5	85.33%	88.11%	78.42%	10	5	0	15
r10g5	84.69%	87.70%	77.34%	10	0	5	15
r5m5g5	85.01%	88.62%	77.69%	5	5	5	15
r10m5g5	85.33%	88.48%	77.92%	10	5	5	20
r15m5g5	85.24%	88.27%	77.80%	15	5	5	25

6. DATA AVAIBILITY

Code for building, training and testing all models are available on GitHub at https://github.com/Elia503/SDE. Trained models are available in a Google Drive folder at https://drive.google.com/drive/folders/1PNbhIzFL2oofV1i6WlRdX8J_47uuSQLD?usp=sharing. Dataset is not yet available because it is protected by copyright laws.

7. REFERENCES

[1] Jorgensen, S. J., Micheli, F., White, T. D., Houtan, K. S. V., Alfaro-Shigueto, J., Andrzejaczek, S., Arnoldi, N. S., Baum, J. K., Block, B., Britten, G. L., Butner, C., Caballero, S., Cardeñosa, D., Chapple, T. K., Clarke, S., Cortés, E., Dulvy, N. K., Fowler, S., Gallagher, A. J., … Ferretti, F. (2022, February 28). *Emergent research and priorities for shark and ray conservation*. Endangered Species Research. Retrieved September 6, 2022, from https://www.int-res.com/abstracts/esr/v47/p171-203/

[2] Baum, J.K., Blanchard, W., 2010. Inferring shark population trends from generalized linear mixed models of pelagic longline catch and effort data. Fish. Res. 102 (3), 229–239. https://doi.org/10.1016/j.fishres.2009.11.006.

[3] Ferretti et al., unpublished data.

[4] Jenrette, J., Liu, Z. Y.-C., Chimote, P., Hastie, T., Fox, E., & Ferretti, F. (2022, May 16). *Shark detection and classification with Machine Learning*. Ecological Informatics. Retrieved September 6, 2022, from https://www.sciencedirect.com/science/article/abs/pii/S1574954122001236

[5] Nanni, L., Lumini, A., Ghidoni, S., & Maguolo, G. (2020, March 14). *Stochastic selection of activation layers for convolutional neural networks*. MDPI. Retrieved September 6, 2022, from https://www.mdpi.com/1424-8220/20/6/1626/htm

[6] Yu, H., Chen, C., Du, X., Li, Y., Rashwan, A., Hou, L., Li, J., 2020. *TensorFlow Model Garden. December 2021.* https://github.com/tensorflow/models.

[7] Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Zitnick, C. L., 2014. *Microsoft CoCo: common objects in context. European conference on computer vision*.https://doi.org/10.48550/arXiv.1405.0312

[8] Ren, S., He, K., Girshick, R., Sun, J., 2016. Faster R-CNN: *towards real-time object detection with region proposal networks. CoRR*. https://doi.org/10.48550/arXiv.1506.01497.

[9] LeCun, Y., Bengio, Y., 1998. *Convolutional Networks for Images, Speech, and Time Series. In the Handbook of Brain Theory and Neural Networks*. MIT Press, Cambridge, MA, USA, pp. 255–258. May 2022. https://dl.acm.org/doi/10.5555/303568.303704.

[10] Tabak, M.A., Norouzzadeh, M.S., Wolfson, D.W., Sweeney, S.J., Vercauteren, K.C., Snow, N.P., Miller, R. S., 2019. *Machine learning to classify animal species in camera trap images: Applications in ecology. Methods* Ecol. Evol. 10 (4), 585–590 https://doi.org/10.1111/2041-210X.13120.

[11] Taylor, L., Nitschke, G., 2017. *Improving deep learning using generic data augmentation*. CoRR. abs/1708.06020. https://doi.org/10.48550/arXiv.1708.06020.

[12] Bridle, J.S., 1990. *Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition*. In: Souli'e, F.F., H'erault, J. (Eds.), Neurocomputing. Springer, Berlin Heidelberg, pp. 227–236. https://doi.org/10.1007/978-3-642-76153-9.

[13] Deng, J., Dong, W., Socher, R., Li, L. J., Li, K., & Fei-Fei, L. (2009, June). *Imagenet: A large scale hierarchical image database*. In *2009 IEEE conference on computer vision and pattern recognition* (pp. 248-255). https://ieeexplore.ieee.org/abstract/document/5206848

[14] Sutskever, I., Martens, J., Dahl, G., & Hinton, G. (2013, May 26). *On the importance of initialization and momentum in Deep Learning*. PMLR. Retrieved September 7, 2022, http://proceedings.mlr.press/v28/sutskever13.html

[15] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). *You only look once: Unified, real-time object detection*. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 779-788).

[16] He, K., Zhang, X., Ren, S., & Sun, J. (2016). *Deep residual learning for image recognition.* In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778).

[17] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. C. (2018). *Mobilenetv2: Inverted residuals and linear bottlenecks*. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4510-4520).

[18] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... & Rabinovich, A. (2015). *Going deeper with convolutions*. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1-9).