Journal of
Marine Science and Engineering

MDPI

Article

Classification of Underwater Fish Images and Videos via Very Small Convolutional Neural Networks †

Marius Paraschiv [1], Ricardo Padrino [1], Paolo Casari [2,*], Eyal Bigal [3], Aviad Scheinin [3], Dan Tchernov [3] and Antonio Fernández Anta [1]

[1] IMDEA Networks Institute, 28918 Leganes, Spain; marius.paraschiv@imdea.org (M.P.); ricardo.padrino@imdea.org (R.P.); antonio.fernandez@imdea.org (A.F.A.)
[2] Department of Information Engineering and Computer Science and CNIT, University of Trento, 38123 Povo, Trento, Italy
[3] Morris Kahn Marine Research Station, Department of Marine Biology, Leon H. Charney School of Marine Sciences, University of Haifa, Mount Carmel, Haifa 3498838, Israel; eyalbigal@gmail.com (E.B.); ascheinin@univ.haifa.ac.il (A.S.); dtchernov@univ.haifa.ac.il (D.T.)
* Correspondence: paolo.casari@unitn.it
† A preliminary version of this work was presented at the IEEE/MTS OCEANS 2020 conference.

Abstract: The automatic classification of fish species appearing in images and videos from underwater cameras is a challenging task, albeit one with a large potential impact in environment conservation, marine fauna health assessment, and fishing policy. Deep neural network models, such as convolutional neural networks, are a popular solution to image recognition problems. However, such models typically require very large datasets to train millions of model parameters. Because underwater fish image and video datasets are scarce, non-uniform, and often extremely unbalanced, deep neural networks may be inadequately trained, and undergo a much larger risk of overfitting. In this paper, we propose small convolutional neural networks as a practical engineering solution that helps tackle fish image classification. The concept of "small" refers to the number of parameters of the resulting models: smaller models are lighter to run on low-power devices, and drain fewer resources per execution. This is especially relevant for fish recognition systems that run unattended on offshore platforms, often on embedded hardware. Here, established deep neural network models would require too many computational resources. We show that even networks with little more than 12,000 parameters provide an acceptable working degree of accuracy in the classification task (almost 42% for six fish species), even when trained on small and unbalanced datasets. If the fish images come from videos, we augment the data via a low-complexity object tracking algorithm, increasing the accuracy to almost 49% for six fish species. We tested the networks with images obtained from the deployments of an experimental system in the Mediterranean sea, showing a good level of accuracy given the low quality of the dataset.

Keywords: underwater images; fish classification; neural networks; optimized architectures; accuracy; sea trial

Citation: Paraschiv, M.; Padrino, R.; Casari, P.; Bigal, E.; Scheinin, A.; Tchernov, D.; Fernández Anta, A. Classification of Underwater Fish Images and Videos via Very Small Convolutional Neural Networks. *J. Mar. Sci. Eng.* **2022**, *1*, 0. https://doi.org/

Academic Editor: Weicheng Cui

Received: 2 May 2022
Accepted: 23 May 2022
Published:

1. Introduction

Automatic image recognition algorithms can greatly help marine detection, observation, and classification tasks. Typically, these activities entail long-term observations of specific underwater fauna or plant species, and require huge efforts in terms of human personnel, equipment, and ship support. In this respect, statistical image recognition algorithms (especially based on deep learning) may be a key asset to relieve constant human presence in underwater observations [1].

Deep learning is a branch of machine learning that employs deep neural networks (i.e., models having several concatenated neuron layers) for pattern recognition and prediction tasks, such as classification and regression. Deep learning has found many successful

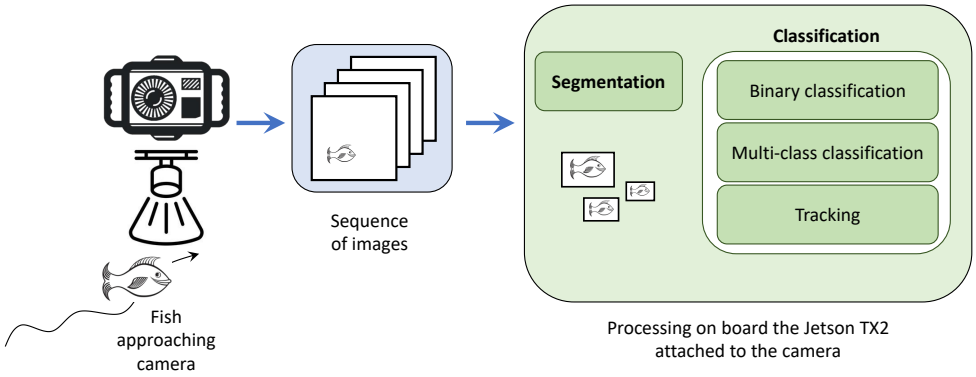


Figure 1. The optical data pipeline, including segmentation, classification, and object tracking.

fields of application, including automated driving [2], medicine [3–6], energy consumption optimization [7], smart agriculture [8], translation among languages [9] as well as between speech and text [10], speech synthesis [11], and object detection [12], among many others. We refer the interested reader to the detailed reviews of the deep learning field in [13,14].

In this paper, we describe our experience with the design and optimization of simplified deep learning models for the identification of fish species appearing in images from submerged cameras. We carried out this work in the context of the Symbiosis project,[1] which developed automatic tools to detect, localize, capture on camera, and classify pelagic fish in the Mediterranean Sea and in the Red Sea.

1.1. Context: The Symbiosis Project

The Symbiosis platform is a set of acoustic and optical subsystems installed along a partially articulated steel pole structure, and designed to be lowered beneath the sea surface as part of a mooring [15]. With reference to Figure 1, the system employs sonar signal processing to detect fish approaching the platform. As one or more specimens enter the field of view of the underwater cameras, the cameras activate to capture images. After segmenting these images to extract detected fish specimens, each resulting segment is automatically classified. Real-time statistics and aggregated results are both stored locally and sent to an external observation station.

Because the Symbiosis platform operates in real time, a number of system design constraints apply. Underwater cameras mounted on the platform have a range of about 10 m. Considering that the movement speed of pelagic fish (including some of the species of specific interest in Symbiosis) can be significantly fast, the complexity of the inference task must be minimal, so that the task completes in reasonable time. Secondly, fish specimens appear with different sizes and at different locations on a captured image, depending on their distance from the camera and on the angle at which they approach it. Hence, we need to preprocess all images to segment areas of interest, crop them and scale them to fit the required input of our classification models. Such preprocessing takes place on the same embedded computer that runs the classification models, and restricts the size of any deep neural network models deployed on the platform. Energy conservation issues are also extremely relevant in this scenario: the underwater platform should ideally run unattended for the longest possible time span. For these reasons, our experience showed that it is not convenient to run very deep neural network models with many layers and weights.

In the next subsections, we first review relevant related work, and then proceed to explain the main ideas, challenges, and contributions of our work.

1.2. Related Work on Deep Learning for Underwater Image Recognition

The research on image recognition applied to the underwater world has been tapping on the similar techniques as those employed for image recognition in other contexts. In particular, convolutional neural layers are a key part of deep learning models for image

recognition and classification. The convolutional layer is especially significant for image processing and classification tasks. Drawing inspiration from the human visual cortex, convolutional layers exploit several invariances (e.g., to local deformation or translation), in order efficiently extract and aggregate features in the data. Moreover, a convolutional layer drastically reduces the number of parameters in a deep neural model, especially compared to multi-layer perceptrons or, in general, fully-connected layers [16].

Iqbal et al. [17] propose a deep neural network model for fish classification that employs progressively smaller convolutional layers with a decreasing stride, followed by two large fully-connected layers. They observe that adding dropout layers increases the performance of their classification model.

Along the same line, Cui et al. [18] propose a deep neural network model based on a sequence of convolutional layers and fully connected layers. Convolutional layers use local features to reduce complexity. The network is trained to recognize fish shapes, and mark them with a bounding box in video frames. As the model should run on embedded computers within autonomous underwater vehicles (AUVs), the authors investigate the effect of data augmentation in the training phase and network simplification.

Deep learning models are also included in some smartphone applications in this domain, e.g., to help identify fish catch and inform about regulations in the catching area [19,20].

Yang et al. [1] review deep learning approaches in the field of aquaculture. Among the advantages of deep learning with respect to expert systems with manually-engineered features, the authors emphasize the capability to automatically extract relevant image features. Moreover, they identify three future trends, namely (i) detailed fish classification, including fish diseases and misbehaviors, (ii) dataset availability, and (iii) advanced models combining deep learning with handcrafted features, and accounting for spatio-temporal sequences.

Classical machine learning algorithms are employed in [21] to detect fish. In order to overcome the limitations of stereo visual systems, the authors propose a two-phase processing scheme. First, they reduce the dimensionality of the dataset via, e.g., principal component analysis (PCA), which results in a more distilled correspondence between pixel-level characteristics and fish shapes. After identifying the shape of a fish, the scheme typically proceeds by segmenting the head of the fish and by recognizing the remaining body objects with pixel or sub-pixel accuracy.

Other efforts rely even stronger on convolutional neural processing. For example, the authors of [22] propose Fast R-CNN (for Regions with CNN), a CNN-based algorithm for fish identification in images. The algorithm proceeds by first generating candidate image segments, that are later combined into larger regions using a greedy algorithm. These segments are finally processed to propose final candidate image regions that likely contain fish specimens. The authors report a mean average precision (mAP) of 81.4% using a dataset of 24k fish images from twelve species.

An end-to-end convolutional network is one of the two methods proposed in [23] to differentiate fish. The method is compared against a scheme built around the Histogram of Oriented Gradients (HoG) technique that extracts object features and inputs them to a classifier, such as a support vector machine (SVM).

The automatic recognition of fish and underwater fauna in general helps underwater research in several ways. For instance, the Fish4Knowledge project [24] focuses on preprocessing underwater camera images, in order to extract information of value to biologists and oceanographers. This helps reduce the otherwise very sizeable data that always-on underwater cameras would produce. Automatic image recognition in Fish4Knowledge is one way to automatically interpret images and store only significant data to be queried later, e.g., to infer the behavior of a given fish species in the environment being monitored.

The relatively more recent Fishial.ai project [25] also targets the automatic fish detection and classification using deep neural networks, with the ultimate objective to assist environmental conservation efforts. At its current stage, the project is calling for contribu-

tions from anyone who can upload tagged fish images, in any environment, both under and above the water.

Earlier works such as [26] employ denoising schemes and the mean shift algorithm to clean an image and split it into subregions. Multiple subregions are then combined into objects using statistical methods.

1.3. Summary of Our Main Ideas

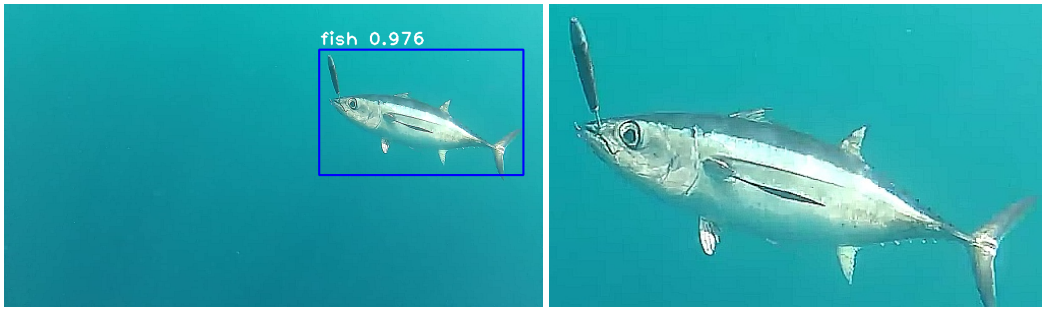
For the reasons explained in Section 1.1 and given the state of the art described in Section 1.2, we focus our effort on the engineering and evaluation of small deep learning models, that preferably incorporate fewer than 20,000 parameters, or up to three orders of magnitude less than state-of-the-art deep learning models for image classification. This significantly reduces the complexity of the models, but at the same time it entails a number of challenges. First, the representation capacity of small models is limited. Second, any dataset collection issues resulting in a low-image quality or unbalanced dataset may compound and lead to poor classification performance. Therefore, different kinds of small models need to be tested, in order to single out the best neural architecture that still offers a workable accuracy while keeping complexity at bay.

Besides the neural architecture, one significant degree of freedom for the classification system design relates to the source of fish images, which often come from video frames. By being able to track a fish specimen across subsequent frames, we can associate multiple segments to the same individual, and can implement a voting procedure to remedy otherwise error-prone one-shot classification outcomes. In practice, we ultimately increase the amount of information presented to the classifier by preparing multiple images that we believe depict the same subject, such that we can increase the likelihood that the classifier makes a correct prediction.

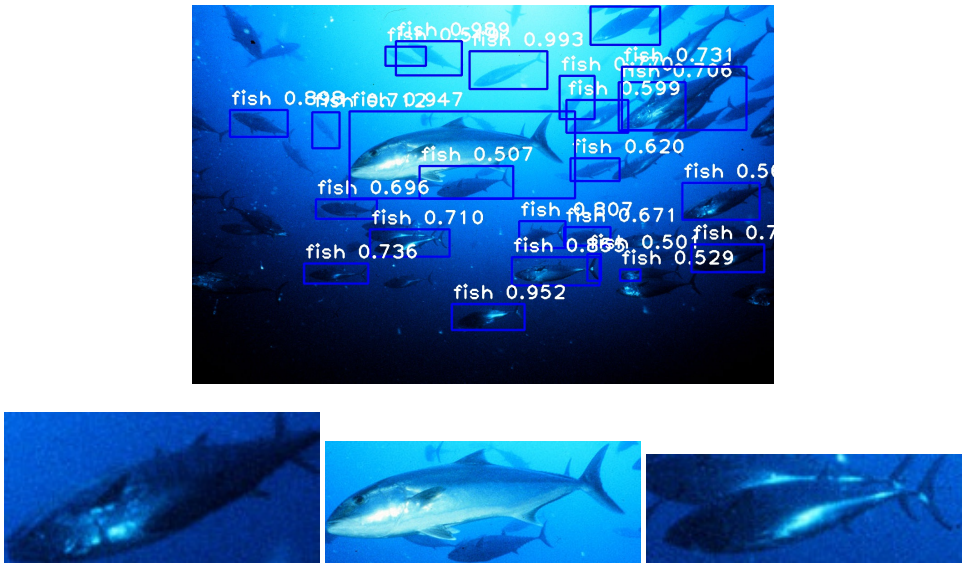
In this study, we target six species of pelagic fish (see Section 4.1). Our objectives are (i) to differentiate between images containing fish from these six species and images showing different species (*binary classification*); and (ii) to tell which among the six target Symbiosis species a captured image represents (*multi-class classification*). We train all classifiers offline, and rather focus on reducing their size and complexity, as such systems typically need to run unattended in the wild for the longest possible time.

1.4. Challenges Related to Datasets

A considerable challenge, in our work, is to produce relevant datasets containing images of Symbiosis's target pelagic fish species. While many fish image and video repositories exist, most scenes are captured outside the water, and portray additional subjects (e.g., a person fishing) or non-natural backgrounds (e.g., the floor of a sailing ship or fishing boat). We chose to avoid such scenes, as they do not conform to the operational environment of the Symbiosis platform. Hence, we decided to use only observation photos and videos, showing the fish specimens swimming free underwater (see Figure 2). Unfortunately, there exists a limited number of such images and videos, since the target specimens usually live in comparatively deep waters. Multiple fish images can be extracted from each video: the drawback is that these images will typically show a few specimens, possibly grouped together, and repeated across several subsequent frames from the same video (see Figure 3). These issues make it challenging to train our deep neural networks. In order to mitigate the limited variety in the dataset and the presence of similar individuals, we have proceeded along the following two directions: (1) we have removed very similar fish images, (2) we employ images taken from different videos when setting up the training and test datasets.



(**a**) Dataset image showing a *Thunnus alalunga* (left: full image; right: segmented fish)



(**b**) Dataset image with several segments portraying *Seriola dumerili* specimens

Figure 2. Images and segments from the dataset used in this paper. (**a**) Image of a *Thunnus alalunga* and the segment extracted from it. (**b**) Image of *Seriola dumerili* and segments extracted from it. Photo credit: Wikipedia.

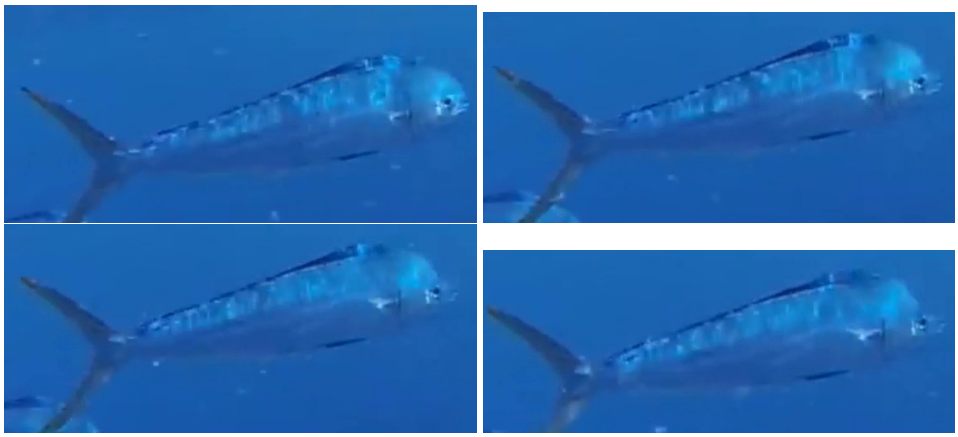


Figure 3. Segments from a *Coryphaena hippurus* extracted from consecutive frames of a video.

1.5. Our Contributions and Main Results

The main contributions of this paper are:

- the engineering and evaluation of small neural network models that can be implemented with constrained embedded platforms;

- a training method that works with small and unbalanced datasets, often taken from similar images extracted from videos;
- a method that assists the decisions of the classifier through object tracking.

With reference to Figure 1, image processing in the Symbiosis platform has the objective to (i) automatically detect fish in a sequence of video frames and segment them, and (ii) to classify the fish segments and predict the species of the specimens therein. For both problems, we employ convolutional neural networks. The former problem, image detection and segmentation, takes place through a method independently developed, which uses a specifically trained instance of the neural network RetinaNET [27,28].

Then, the latter problem, image classification, which is the focus of this paper, exploits different convolutional architectures to identify the segments that contain one of the six pelagic fish species of interest for Symbiosis. The first step to achieve this is to tell apart those segments containing irrelevant fishes or non-fish artefacts (binary classification), which is followed by a second step in which individual fish species are determined (multi-class classification). For comparison purposes, we consider image classification with several popular convolutional architectures, such as VGG16 [29], François Chollet's architecture [30], as well as the more recent RBB [31] and MobileNetV3 [32] networks. The latter has been specifically designed with the computation capabilities of mobile devices in mind. With respect to the above works, in this paper we advocate the use of much smaller neural network models, with fewer than 20,000 weights. These models can run in much shorter time than the deep convolutional networks found in popular image recognition models such as RetinaNet [27] and VGG [29], and thus require much less energy. This aspect is very relevant for embedded systems that run unattended in the wild. Moreover, we show that tracking objects across multiple images or video frames increases the classification accuracy by up to 7%. We thus advocate this method as a promising alternative for those cases where limited training data may reduce classification accuracy.

A preliminary version of this paper discussing offline training issues was presented at the IEEE/MTS Global OCEANS 2020 Conference [33].

The structure of the paper is as follows. Section 2 discusses theoretical aspects, including a brief overview of the neural architectures we employ in this work, the object tracking method, and the components of our data pipeline. Section 3 presents experimental results both for binary classification and for multi-class classification, including the impact of object tracking on the accuracy of the classification task. In Section 4 we describe the actual Symbiosis system, and present the results of two sea trials performed with the Symbiosis platform in deep and shallow Eastern Mediterranean Sea waters. Finally, Section 5 concludes the paper.

2. Theoretical Aspects

The above literature review confirms that, with the exception of a few works, the greatest majority of recent image classification approaches employ a form of convolutional neural networks, either in combination with classical techniques (which carry out feature extraction, denoising, or other preprocessing steps before the convolutional network classifies the input based on extracted features), or in an end-to-end manner (where the convolutional network is designed to directly operate on the input image, extracting relevant features, and complete the classification task). More broadly, deep learning algorithms have replaced hard-coded approaches as a de-facto standard for vision tasks, thanks to their relatively easy domain adaptation and to the limited complexity of the model generation steps.

In our work, we focus on two kinds of benchmarking tasks: (i) binary classification, to tell apart the six species of interest of Symbiosis from other species; and (ii) multi-class classification, to distinguish specimens belonging to each of the six Symbiosis species. To make the most of hardware limitations on embedded platforms, we elect to use small neural architectures with between $10,000$ and $20,000$ trainable parameters. As shown in Figure 1, the classification task proceeds through two subsequent steps: image classification (either binary or multi-class) and object tracking. For tracking, we use the SORT [34] algorithm. We

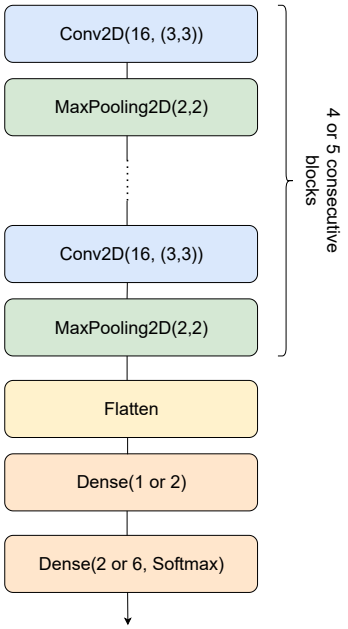


Figure 4. A general view of the neural network architectures we employ for binary and multi-class classification.

exploit tracking to recognize images that represent the same fish, typically across multiple subsequent frames of a video. These images, once segmented, are classified separately. The output of the classifier is a vector of probabilities for each image, that describes how confidently the classifier predicts the image to pertain to one of seven classes: one class for each the six Symbiosis species, plus one additional class for any *other species* not of interest. A voting system then decides on the accepted result, and assigns a final label to each detected object.

The centerpiece of the pipeline are the proposed small architectures. They are essentially variations on the same structure, as seen in Figure 4, starting with an alternation of 2D convolutions followed by MaxPooling layers (forming the feature extraction component of the network) and finishing with a set of dense layers (composing the classifier part). In terms of nomenclature, for the scope of readability, we chose to name the networks based on their structure. As such, $4(C_{16}M)D_8D_{16}D_y$ refers to 4 initial blocks of 2D convolutions with 16 filters followed by MaxPooling, and a classifier component formed by a dense layer of 8 units, one of 16 units, and a final dense output of y units, where y is the number of classes of the particular experiment.

We compare our architectures against well-known networks, namely VGG16 [29], a baseline convolutional architecture proposed as part of the Keras library tutorial, which we chose to denote as the *Chollet* architecture [30] (after the initiator of Keras), the RBB network [31], and the MobileNetV3 architecture [32]. The RBB network consists of two convolutional input layers, followed by 16 residual blocks, an average pooling as well as a 1056-size fully-connected layer. The residual blocks are composed of 3 convolutional layers and a skip-connection. In order to exploit the complexity of the learned representations, RBB increases the number of convolutional filters in the residual blocks, with the network depth. A result of this architecture is the very large number of parameters. The MobileNetV3 networks are bottleneck convolutional networks (i.e., where the size of the representation decreases by proceeding deeper through the network) specifically designed to be applied on mobile devices, under memory and computational constraints. Two similar architectures of different size have been proposed, dubbed "Large" and "Small" in this paper. A detailed description of the networks is provided in [32] (Tables 1 and 2). The RBB and MobileNetV3 networks have been specifically proposed for mobile deployments in the literature.

Table 1. Structure of the architectures considered in this paper, along with their total number of trainable parameters. The final dense layer D_y is either D_2 or D_6, for binary and multi-class classification, respectively. We remark that we used a pre-trained version of VGG16, where we only re-trained the final layer of the network to match our dataset.

Architecture	Number of Parameters
$4(C_{16}M)D_8D_{16}D_y$	18,030
$4(C_{16}M)D_{32}D_{32}D_y$	50,166
$5(C_{16}M)D_8D_y$	10,942
$5(C_{16}M)D_{32}D_y$	14,566
$5(C_{16}M)D_{16}D_8D_y$	12,238
VGG16	15,124,518
Chollet	1,667,494
RBB network	69,077,990
MobileNetV3 Large	8,739,558
MobileNetV3 Small	3,035,174

All architectures discussed in the present paper are listed, together with their number of trainable parameters, in Table 1. The proposed architectures have between 10 and 20,000 parameters, with one exception of roughly 50,000 parameters. In all cases these numbers are considerably smaller than the 1.6 million parameters of Chollet, the 3 to 8.7 million parameters of MobileNetV3, the over 15 million parameters of VGG16, and the almost 70 million of RBB. In particular, we observe that the RBB network is three orders of magnitude larger than the networks we propose, and hence will not be considered as a practical option. The MobileNetV3 networks also have two orders of magnitude more parameters than our networks.

3. Experimental Results

3.1. Image Dataset Creation

Neural networks require a large set of training examples. This is especially true for the deep architectures employed for image processing and recognition. In the absence of a sufficiently rich dataset, the many trainable parameters (or weights) of the network converge to values that overfit the limited number of training examples provided, so that the model becomes unable to generalize to different, previously unseen images [35]. Moreover, for classification problems, the training set should representing the different classes uniformly, so that the neural network does not "specialize" on a specific class while remaining unable to recognize the others. In Symbiosis, one of the most challenging tasks was precisely to compile a reasonably large and curated training dataset. This required us to collect, clean and aggregate a sufficiently large quantity of images in a homogeneous form, along with their corresponding relevant meta-data: labels, bounding boxes (regions of interest, or ROI), etc.

Symbiosis focuses on the recognition of images portraying one of six species of bony fishes (see Figure 5), selected based on their commercial value, their behavior (e.g., schooling or not schooling), appearance, as well as occurrence at the foreseen deployment locations in the Eastern Mediterranean Sea and in the Red Sea. These species of interest are:

- Albacore tuna (*Thunnus alalunga*).
- Greater amberjack (*Seriola dumerili*).
- Atlantic mackerel (*Scomber scombrus*).
- Dorado (dolphinfish, *Coryphaena hippurus*).
- Mediterranean horse mackerel (*Trachurus mediterraneus*).
- Swordfish (*Xiphias gladius*).

The Biology team of the Symbiosis project helped compile a first dataset containing raw data from multitude of private and public data sources. This included underwater videos and photographs of the above six species. Each item was manually revised and

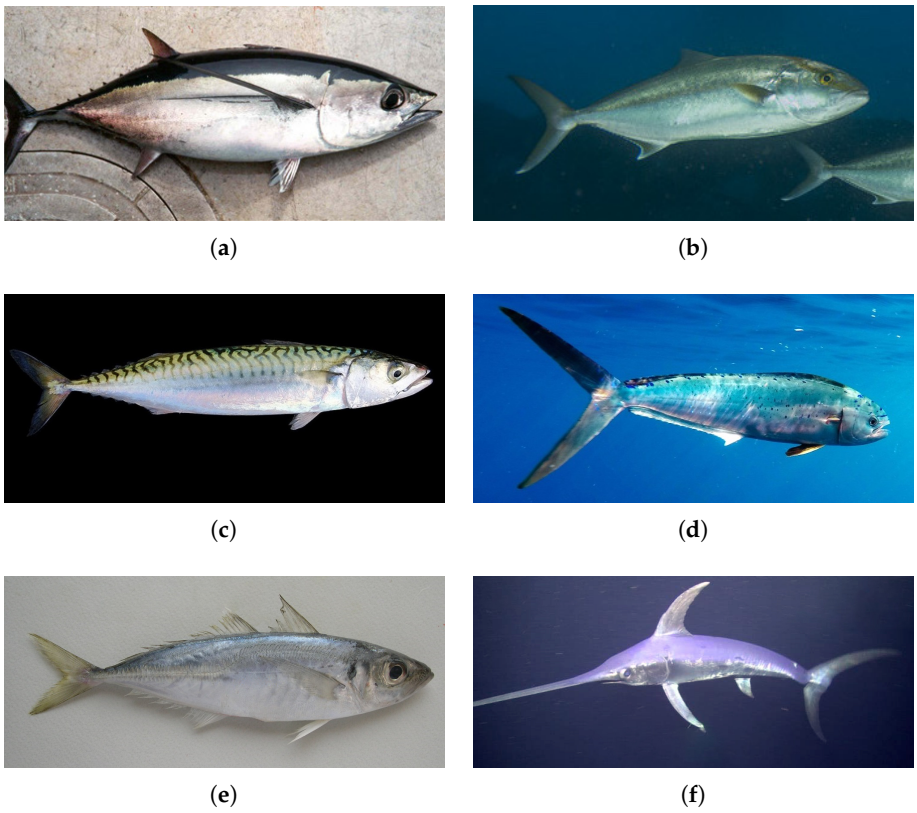


Figure 5. The six species of interest for the Symbiosis project considered in this work. (**a**) Albacore tuna (*Thunnus alalunga*). Photo credit: User:Pvmoutside, Public domain, via Wikimedia Commons. (**b**) Greater amberjack (*Seriola dumerili*). Photo credit: H. Nativ, Morris Kahn Marine Research Station. (**c**) Atlantic mackerel (*Scomber scombrus*). Photo credit: Hans Hillewaert, licensed with CC BY-NC-ND 2.0. (**d**) Dorado (*Coryphaena hippurus*). Photo credit: Uri Magnus Dotan. (**e**) Mediterranean horse mackerel (*Trachurus mediterraneus*). Photo credit: Fisheries Research Institute, CC BY-SA 4.0, via Wikimedia Commons. (**f**) Swordfish (*Xiphias gladius*). Photo credit: NOAA Deep Sea Coral Research & Technology Program—Pelagic Research Services, Public domain, via Wikimedia Commons.

labeled as one of the six species, and those images and videos that included several different species were discarded in this phase to help dataset labeling. After extracting frames from the videos, we formed a first collection of training samples by segmenting the extracted frames and images through the algorithm in [28]. This produced about 1.5 million image segments, each containing a region-of-interest (ROI) with a potential fish. As the algorithm was still being optimized at the time, we proceeded by inspecting all images manually to discard those that did not contain a fish, or those where the size or quality of the fish image was too low, leaning itself to misinterpretation even by a human. This filtering process was based on such fish body features as minimum size, as well as image quality features such as full or partial fish body coverage, general image quality, and the detection accuracy reported by the algorithm in [28]. Initially, we crowdsourced part of the process. However, the majority of the image quality assurance and validation process was completed manually by the Symbiosis team, and yielded 51,260 validated segments. It is worth mentioning that some of the segments included in the data set would not allow a human expert to identify the species. They have been maintained in the hope that the neural networks are able to extract and use features that humans cannot, especially when an object tracking process presents a sequence of images of the same object to the classification algorithm.

We employed the Symbiosis dataset for both classification tasks of interest in this work: binary classification (meaning that the algorithm should distinguish between our species of interest, grouped into two classes), and multi-species classification (or distinguishing

Table 2. Distribution of images subsets for training. Total number of images: 51,260. The worst-case imbalance is almost 8:1 between *Seriola dumerili* and *Trachurus mediterraneus*.

Species	Training	Validation	Testing
Thunnus alalunga	4100	1393	1353
Seriola dumerili	8563	2931	2803
Scomber scombrus	2679	840	812
Coryphaena hippurus	7631	2560	2576
Trachurus mediterraneus	1884	654	680
Xiphias gladius	5789	2035	1977
Total	30,646	10,413	10,201

among our six species of interest). For this purpose, we split the dataset into train, validation and test subsets, containing a distribution of (60%, 20%, 20%) of the dataset related to each species. It is worth noting that we kept segments from the same videos together in the same set during this process. This data organization further helps limit overfitting by avoiding that very similar images from the same video spread from the training set to the validation and test sets. The size of the Symbiosis dataset for each of the six Symbiosis species appears in Table 2. We observe that the total number of images is not exceedingly small, yet these images are very unevenly distributed among the six species, reaching a maximum coverage of *Seriola dumerili* with about 14,300 images, and a minimum coverage of *Trachurus mediterraneus*, with about 3200 images.

3.2. Classification Process

As mentioned above (see also Figure 1), our optical detection and classification process consists of two separate stages that rely on convolutional neural networks: (1) detection and segmentation, that relies on a retrained version of RetinaNET [27] to identify areas corresponding to fish images in a raw image or video frame [28]; and (2) classification, where each segment is processed to determine if it portrays one of the six species of interest, and which one. In this paper, we focus on the engineering of light architectures with the aim to provide results comparable to well-known networks (such as VGG16 [29] or the Chollet architecture [30]).

We recall that all architectures we engineer in this paper have a similar structure, namely a sequence of pairs of layers, where each pair contains a convolutional layer followed by a max pooling layer (denoted C_{16} and M, respectively). After these pairs, comes a sequence of dense layers (denoted D_x, $x \in \{8, 16, 32\}$), and a final classification layer (D_y, $y \in \{2, 6\}$, see Figure 4). We remark that we leverage the pre-trained version of VGG16, and rather only re-train the final layer of the architecture using our particular dataset.

When images are segmented from subsequent video frames, tracking becomes an option to help the classification task. Our tracking component exploits SORT, an open-source tracking algorithm that performs well in the presence of typical fish mobility [36–38]. We did not build a third deep learning model for this purpose, in order to keep the computational complexity low, and thus facilitate the deployment of the algorithm on constrained embedded systems. The tracker receives image segment coordinates from every subsequent frame, that correspond to two opposite corners of each ROI. It then provides an artefact identifier and the predicted position of the object, in the next frame. We then compare the predicted coordinates with the actual coordinates of the artifact in a subsequent frame by computing the ratio between the intersection and the union of two segments, i.e., the intersection over union (IoU) metric. For those segment pairs where the metric is maximum and exceeds a predefined threshold, we declare the artifacts to represent the same subject across subsequent frames. It is worth mentioning that the tracker updates its predictions on a per-frame basis, even when the tracked object is missing in some frame. This is particularly useful if the detection and segmentation stage outputs a

Table 3. Accuracy of the **binary** classification task with and without the tracking component. The *Mean*, *Max* and *Voting* columns refer to the type of decision scheme used when the tracking mechanism is applied. The dataset is divided into train, validation, and test subsets as shown in Table 2, but with the species aggregated into two classes: (1) *Seriola dumerili* and *Coryphaena hippurus*, and (2) *Thunnus alalunga*, *Xiphias gladius*, *Scomber scombrus* and *Trachurus mediterraneus*. The results shown are the mean of 3 realizations.

Architecture	No Tracking	Mean	Max	Voting
$4(C_{16}M)D_8D_{16}D_2$	61.55%	61.05%	61.67%	61.08%
$4(C_{16}M)D_{32}D_{32}D_2$	58.35%	60.51%	60.53%	60.71%
$5(C_{16}M)D_8D_2$	62.08%	64.85%	63.80%	64.39%
$5(C_{16}M)D_{32}D_2$	60.36%	63.43%	62.24%	63.52%
$5(C_{16}M)D_{16}D_8D_2$	58.88%	60.47%	60.20%	60.46%
VGG16	73.94%	80.47%	73.86%	79.95%
Chollet	60.66%	65.42%	68.39%	65.01%

false negative in an intermediate frame, something that is likely to happen in videos with a low frame rate.

3.3. Classification Results with and without Tracking

To evaluate our neural network architectures, we have produced two versions of the Symbiosis dataset: in the first version, we group the six species of interest into two classes, one including all instances of *Seriola dumerili* and *Coryphaena hippurus*, and another class grouping the remaining species; in the second version, we keep the classes separate, such that each class represents a different species. We then employ the first dataset for binary classification, and the second for multi-class classification, respectively. We remark that the above binary classification experiment is slightly different from the focus of Symbiosis, which would require us to differentiate between the species of interest for Symbiosis and all other species. However, performing the above experiment allows us to (i) rely on a sufficiently balanced dataset; (ii) train and test our networks using data taken in a uniform environment; (iii) employ images validated by experts; and (iv) evaluate the impact of tracking on the quality of the classification. Later on, in Section 3.4, we will instead comment on the results of a proper binary classification experiment, that distinguishes Symbiosis species from other species depicted in independently collected photo datasets.

The setup for our experiments is identical in both cases: segmented images are first sent to a classifier, in order to establish the fish species. A group of segmented images, along with their class, are combined via tracking, in order to determine if the segments represent the same object. In the presence of a positive tracking output, that ties multiple images to the same subject, we place classification results into a vector, and use one of three methods to finally identify the fish species: by taking the mean of all values (denoted as *Mean*), by taking the maximum value (*Max*) or by using a majority voting scheme (*Voting*). We then employ the final classification result to compute the accuracy of our algorithm.

We present the results of binary classification in Table 3. We observe that the VGG16 architecture achieves the best accuracy (about 74%) without tracking. However, this comes at the cost of a complex deep neural network, having more than 15 million parameters, and requiring significant computational power on embedded devices. Conversely, our $5(C_{16}M)D_8D_2$ architecture achieves 62% accuracy with fewer than 11,000 parameters. A second observation from Table 3 is that tracking increases the accuracy, in some cases significantly (about 8% for Chollet and up to 4% for our architectures).

Table 4 provides the results of multi-class classification among the six Symbiosis species. Here, VGG16 achieves again the highest accuracy (more that 51% without tracking and almost 56% with tracking), but our much simpler architectures with roughly 14,000 parameters have also a workable accuracy of more than 39% without tracking, and more than 44% with tracking. Like for binary classification, tracking significantly improves accuracy.

Table 4. Accuracy of the **multi-class** classification task with and without the tracking component. The *Mean*, *Max* and *Voting* columns refer to the type of decision scheme used when the tracking mechanism is applied. The dataset is divided into train, validation, and test subsets as shown in Table 2. The results shown are the mean of 3 realizations.

Architecture	No Tracking	Mean	Max	Voting
$4(C_{16}M)D_8D_{16}D_6$	37.97%	40.63%	42.29%	39.55%
$4(C_{16}M)D_{32}D_{32}D_6$	39.47%	42.52%	44.15%	42.60%
$5(C_{16}M)D_8D_6$	37.47%	41.42%	41.17%	41.45%
$5(C_{16}M)D_{32}D_6$	39.17%	41.85%	44.26%	41.57%
$5(C_{16}M)D_{16}D_8D_6$	38.09%	40.17%	40.41%	40.44%
VGG16	51.37%	55.90%	52.03%	55.87%
Chollet	31.85%	32.84%	32.50%	32.95%

In order to compare these results with state-of-the-art mobile networks, we have run the multiclass classification with the MobileNetV3 networks [32]. Interestingly, even with the large number of trainable parameters of these networks, the average accuracy obtained is roughly 25% (24.56% with MobileNetV3 Large and 25.01% with MobileNetV3 Small). This is much lower than the accuracy obtained with the other networks.

It is important to point out that the VGG16 has been pre-trained, and can thus take advantage of previously-learned parameters. Furthermore, relatively small datasets can prevent large architectures from achieving their peak representational capacity, due to overfitting (potentially explaining the relatively poor performance of architectures such as Chollet). In practical applications, data collected from probes such as the one used in the present experiments is typically scarce. As such, small architectures benefit from fast adaptability to small datasets, apart from their very low computational requirements.

3.4. Additional Image Classification Results

We present now additional classification experiments performed with the different architectures presented. These experiments include binary classifications that are similar to the one of the the Symbiosis system, since they classify between the six target fish species and other species. In these experiments the data available does not allow the use of tracking, since the new dataset used does not contain videos, and they only evaluate the performance of the classification process of individual images. The datasets used in these experiments are the following.

- *Symbiosis-NCC dataset*: This dataset contains a subset of the segments in the Symbiosis dataset, in which images are guaranteed to have Normalized Cross-Correlation (NCC) below 50%. The NCC filter applied to the original Symbiosis dataset removes segments of the same fish extracted from the same video that are very similar. The number of images of each species in the Symbiosis-NCC dataset are shown in Table 5. The total number of images is 9262, all of our 6 species of interest.
- *QUT+NOA dataset*: 4372 images of QUT[2] (see also Figure 6) and 749 images of NOAA[3], totaling 5121 images. These are collections of fish images, which do not include videos. They include mostly reef fishes, and do not include images of our 6 species of interest.

With these datasets we have carried out 3 experiments, two of binary classification and one of multi-class classification.

- *Binary–No mix:* In the first binary experiment the two classes are the images from the Symbiosis-NCC and QUT+NOA datasets, respectively. These images are split among training, validation, and test sets in a proportion of roughly (80%, 10%, 10%), but making sure that images from the same video in the Symbiosis dataset are all in one set (see Table 6).
- *Binary–Mix:* In this second binary experiment, the split strategy of the QUT+NOA dataset is the same used in the first one, while the Symbiosis-NCC is split randomly in exactly (80%, 10%, 10%) proportions. This means that images from the same video

Table 5. Number of images per species in the Symbiosis-NCC dataset, obtained from the Symbiosis dataset after filtering in order to have Normalized Cross-Correlation below 50%.

Species	Number
Thunnus alalunga	768
Seriola dumerili	1832
Scomber scombrus	1575
Coryphaena hippurus	3398
Trachurus mediterraneus	628
Xiphias gladius	1061
Total	9262

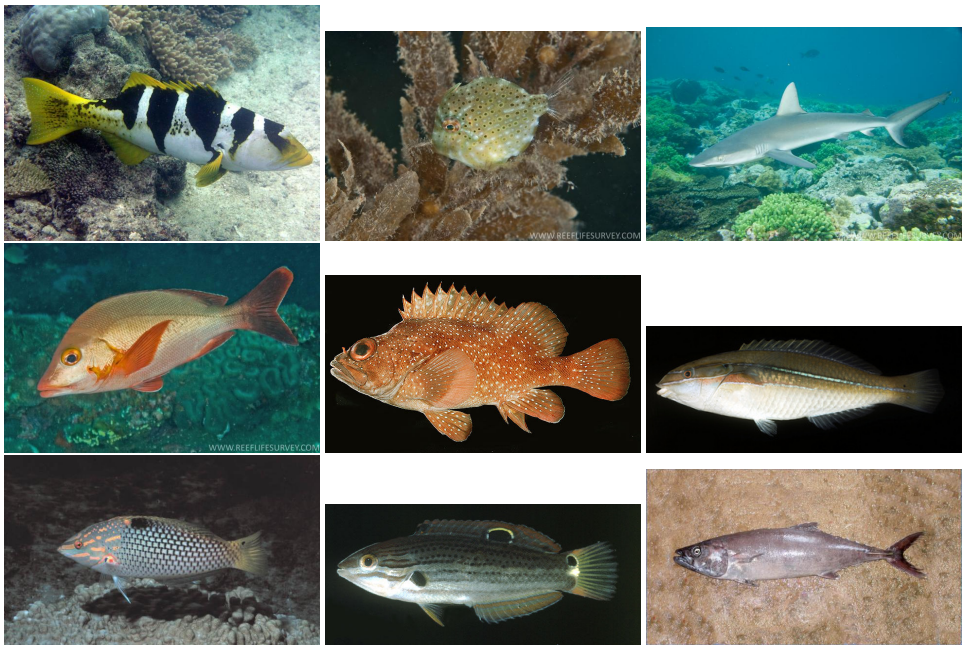


Figure 6. Images from the QUT dataset. Photo credit: QUT FISH Dataset, Fish Species Image Data, Sripaad Srinivasan, Fisheries Research Institute, CC BY-SA 3.0.

may be in the training and test sets. This may help increase the accuracy since these images may be similar (although their NCC is below 50%).

- *Multi-class–Mix:* In the multi-class classification experiment the Symbiosis-NCC dataset is randomly split in exactly (80%, 10%, 10%) proportions, as shown in Table 7. Again, images form the same video can fall in different sets.

Table 8 shows the accuracy of the 3 experiments. All experiments have large accuracy. As can be seen, the *Binary–Mix* experiment has slightly higher accuracy than that of *Binary–No mix*, which seems to indicate that training and testing with images from the same video indeed helps. This is also confirmed by the fact that accuracy of the experiment *Multi-class–Mix* is much higher than the one observed in the previous section, Table 4. One important remark that must be made here is that, while training and testing with images extracted from the same video may indeed be helpful to increase the accuracy of the models, care must be taken that the frames representing different examples are sufficiently different to prevent simple memorization. In this work we applied a filter that provided an indication of the level of similarity between two frames, and discarded the ones that were not sufficiently distinguishable. Furthermore there is a risk that the model may learn other background objects that may appear in two images, and use them in the classification process. This problem is partially mitigated by the similarity filter and partially by the

Table 6. Split of the datasets used for the *Binary–No mix* classification of Table 8. In the Symbiosis-NCC dataset, segments from the same video are in the same set (among train, validation and test).

Dataset	Train	Validation	Test	Total
Symbiosis-NCC	7863	676	723	9262
QUT+NOA	4097	512	512	5121

Table 7. Split of the Symbiosis-NCC dataset used for the *Multi-class–Mix* classification of Table 8. Segments from the same video could be placed in several sets of train, validation and test.

Species	Train	Validation	Test	Total
Thunnus alalunga	614	77	77	768
Seriola dumerili	1464	184	184	1832
Scomber scombrus	1260	158	158	1576
Coryphaena hippurus	2718	340	340	3398
Trachurus mediterraneus	502	63	63	628
Xiphias gladius	840	106	106	1052

Table 8. Accuracy of the additional experiments: *Binary–No mix* is a binary classification experiment using the dataset QUT+NOA versus the Symbiosis-NCC dataset of Table 5, with the split of Table 6. *Binary–Mix* is a binary classification experiment using the dataset QUT+NOA versus the Symbiosis-NCC dataset of Table 5, with a (80%, 10%, 10%) split. *Multi-class–Mix* is a multiclass classification experiment using the Symbiosis-NCC dataset of Table 5, with the (80%, 10%, 10%) split of Table 7.

Architecture	Binary No Mix	Binary Mix	Multi-Class Mix
$4(C_{16}M)D_8D_{16}D_y$	92.57%	98.67%	88.57%
$4(C_{16}M)D_{32}D_{32}D_y$	94.68%	98.84%	91.45%
$5(C_{16}M)D_8D_y$	93.60%	98.53%	88.57%
$5(C_{16}M)D_{32}D_y$	94.25%	98.63%	88.97%
$5(C_{16}M)D_{16}D_8D_y$	93.09%	98.40%	88.90%
VGG16	96.41%	99.37%	88.00%
Chollet	94.08%	98.67%	96.12%

object detection and cropping performed prior to classification, a process that removes most of the surrounding background.

It is worth mentioning that the accuracy observed with VGG and Chollet is not significantly higher than the one observed with the small architectures that are proposed in this paper. Yet, all experiments show significantly larger accuracy than those in Table 3. We believe that this is is due to the different setting of the QUT+NOA dataset, as well as to its limited size (about 1/2 of our dataset). Such differences may lead larger networks such as Chollet and VGG-16 to overfit the classification problem, including the chance that classification relies not just on the morphological characteristics of the fish, but also on other image features, including the background. Conversely, our smaller networks have a more limited data representation power, and are less prone to overfitting.

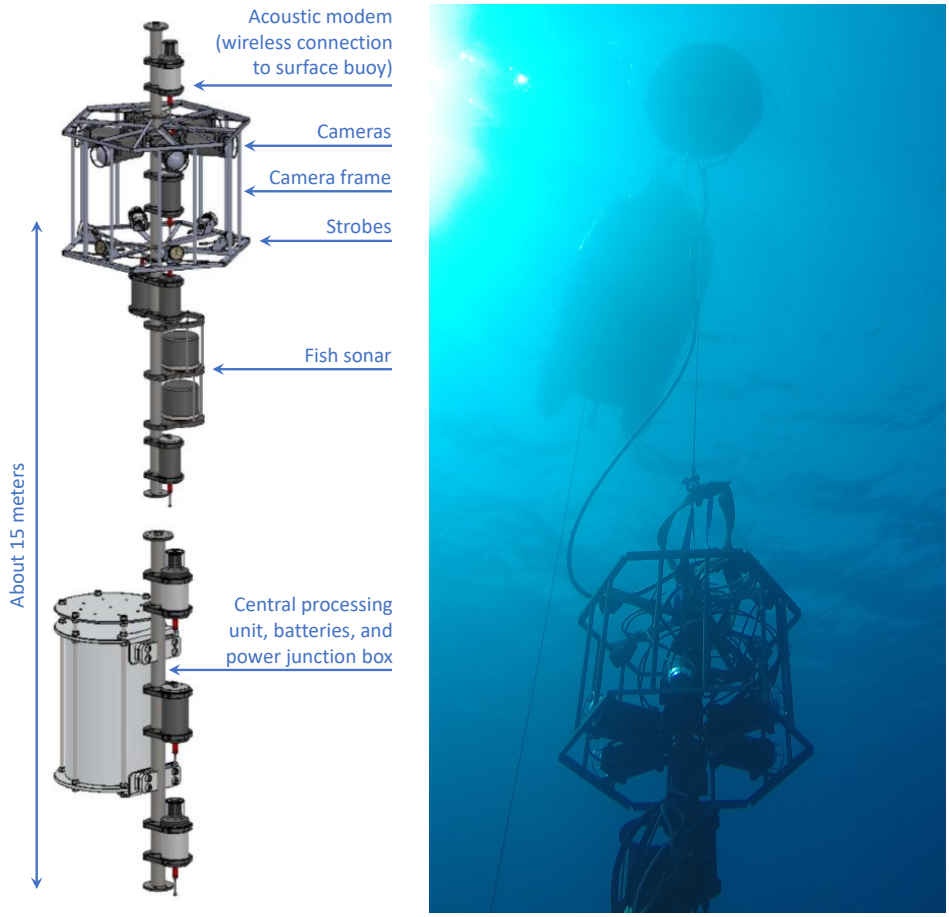
4. Deployment in the Wild

4.1. Experimental Setup

The Symbiosis system is an opto-acoustic system built to detect, track, monitor, and classify pelagic fishes in the wild. The system is designed to be submerged into the sea, attached to a mooring cable, and includes both acoustic and optical subsystems (see Figure 7). The acoustic system is a sonar designed and built ad hoc starting from the joint operation of multiple sub-elements [39,40]. The optical subsystem includes two camera arrays, one in the top-most section and another in the bottom part of the system, about 15 m apart. Each camera array includes six units located at the six vertices of a hexagon,

such that the fields of view of adjacent cameras overlap by 10.5 degrees on each side. The camera frame also mounts six strobe units, one per camera, that help image capture in low-light conditions. Each camera and its strobe are controlled by an NVIDIA Jetson TX2, with an embedded GPU that runs our neural network models for image processing. A large waterproof case hosts a central control and computation unit (another NVIDIA Jetson TX2), as well as batteries to power the whole system. The system can work autonomously under water, at a typical depth of 20 to 40 m. The surface buoy the system is attached to provides connectivity with shore stations or nearby ships through a long-range digital radio.

Energy consumption is one of the most important system constraint for the platform. The organization of the platform's operations follows from this constraint, e.g., by avoiding to operate the acoustic and optical detection subsystems simultaneously. Instead, the acoustic system operates continuously in a low-energy mode, until a possible fish is detected in within 10 to a few hundred meters. At this point, the advanced sonar localizes and tracks the target(s). If the system determines that the trajectory of a fish will lead it at a distance of 1 m to 10 m from one or more cameras, the optical pipeline activates. The camera units capture a sequence of images, and perform the optical detection and classification of the possible fish candidates. The results of the cameras are aggregated in a central unit, that periodically generates detection summaries for offline analysis.



(**a**) Symbiosis buoy scheme (**b**) Red Sea deployment in Eilat, Israel

Figure 7. (**a**) A 3D view of the Symbiosis platform and its main components. The lower section, with its own sonar and camera arrays, is arranged symmetrically with respect to the central section, and is not shown here for brevity. (Adapted from [33].) (**b**) A photo of the platform's deployment in Eilat, Israel (Red Sea), showing the surface buoy, the top camera unit, as well as part of the sonar.

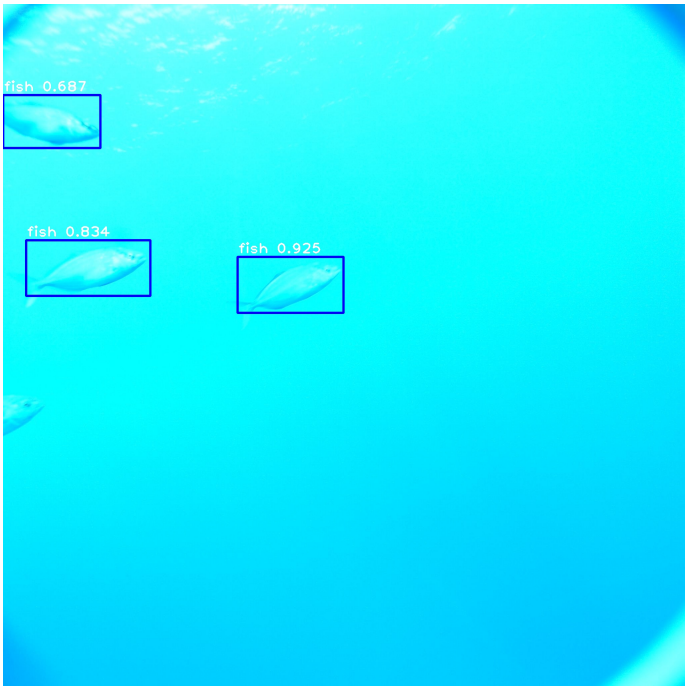
4.2. Themo Deployments

The system described in Section 4.1 has been tested in real deployments in the Mediterranean Sea, on the coast of Israel, in the locations of the deep and shallow moorings of the THEMO[4] system:

- *Deep Themo:* Deployment 55 km west of Haifa, Israel, in a sea area where the water depth is 1400 m. The platform was placed 10 m below the surface and was active for 36 hours.
- *Shallow Themo:* Deployment 11 km from shore, where water depth is 130 m. The system was 5 to 25 m below the surface and was active for 4 weeks.

The Deep Themo deployment captured 39 images, to which the detection and segmentation algorithm was applied (see, e.g., Figures 8 and 9). This produced 55 segments that were classified manually. Out of the 55 segments, 28 actually contained the image of a fish, which Symbiosis biologists then identified. This manual classification was done at two levels:

- Only individual segments were considered for the classification, regardless of the whole image or other images captured. The expert determined the species to which the fish in the segment belongs. The expert labeled the fish segment as *unidentifiable* if the species cannot be recognized. After this process, 21 images were classified by



(**a**) Image with *Seriola dumerili* from Deep Themo

(**b**) Segments with *Seriola dumerili* extracted from the image

Figure 8. (**a**) An image captured in the Deep Themo deployment with 4 specimens of *Seriola dumerili*, out of which 3 have been identified during segmentation. (**b**) Segments of the fishes extracted from the image.

species (20 segments contained *Seriola dumerili* specimens, Figure 8, and one contained a *Coryphaena hippurus*, Figure 9).

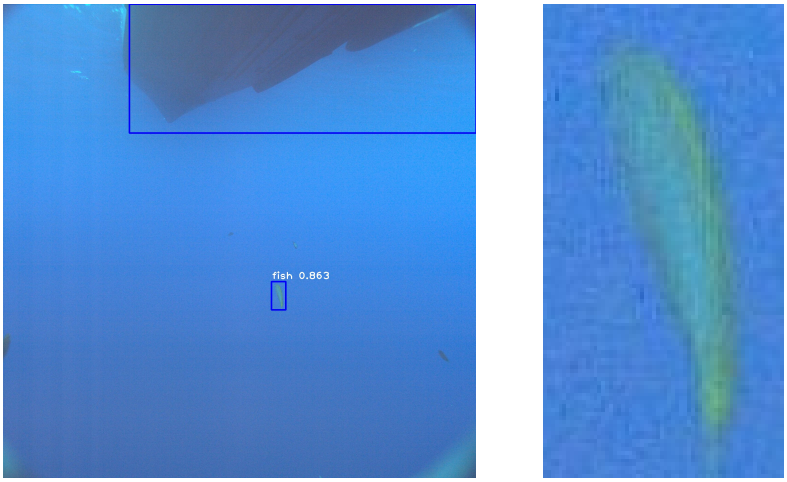
- For the segments labelled as *unidentifiable*, the whole images from where they were extracted were used to help the species classification. This allowed to classify 3 more segments as containing *Seriola dumerili* specimens.

Since the 39 images captures were not obtained in sequence, all the tracking algorithm could do was to validate that there were no two segments that corresponded to the same artifact (fish or not). If we had had images in close sequence these would have been used in a third level of the manual species classification to attempt the classification of the 4 remaining segments labelled *unidentifiable*.

A similar process for the images captured in the Shallow Themo deployment produced 144 fish segments, out of which 115 could be manually classified as *Seriola dumerili* (see Figure 10; observe that some fishes were not detected). Again, the segments in different images did not correspond to the same artifact, so tracking did not help in the classification.

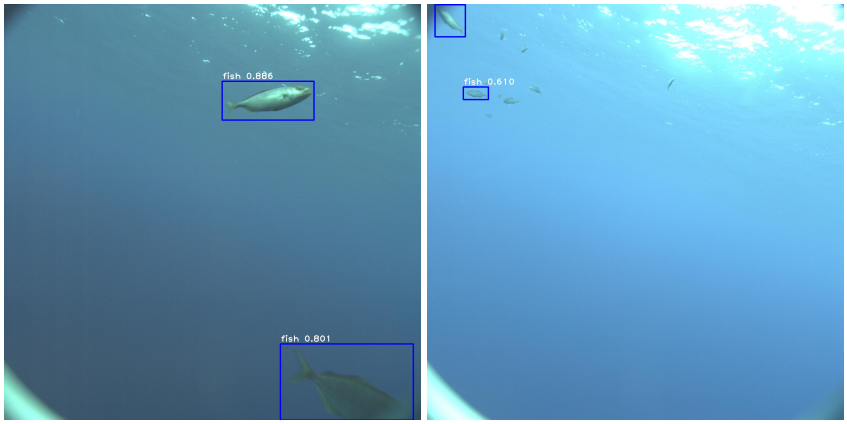
The VGG binary classification applied by the deployed system to all the segments classified manually in both deployments decided that all of them belong to the six species of interest. By processing these segments offline after the deployment, we obtained the same result when performing the binary classification with the small custom architectures proposed here.

The multi-class classification process has been carried out offline to the segments obtained in both deployments. Every architecture has been trained with the images of the Symbiosis dataset presented in Table 2, but the training set used is the union of the training and test sets of Table 2, while the validation set is the same. The results of these classification experiments are shown in Table 9, for all architectures and the 24 Deep Themo and the 115 Shallow Themo segments. As can be observed, the accuracy for Deep Themo is larger than the accuracy for Shallow Themo. In fact, for Deep Themo the VGG architecture has the largest accuracy (more than 86%). However, architectures $4(C_{16}M)D_{32}D_{32}D_6$ and $5(C_{16}M)D_{16}D_8D_6$ show also relatively high accuracy of more than 65%, with many fewer parameters. For Shallow Themo, the accuracy is lower, and there is no architecture that is obviously best that the others. It is worth to note that the test sets used for the results of Table 9 are very small (24 and 115 segments) and extremely biased (all segments are of *Seriola dumerili* except one). This may have influenced significantly the accuracy values obtained. That may explain, for instance, the extremely low accuracy shown by the $4(C_{16}M)D_8D_{16}D_6$ architecture with the Shallow Themo segments. Only VGG16 was able to identify the fish of Figure 9 as *Coryphaena hippurus*.

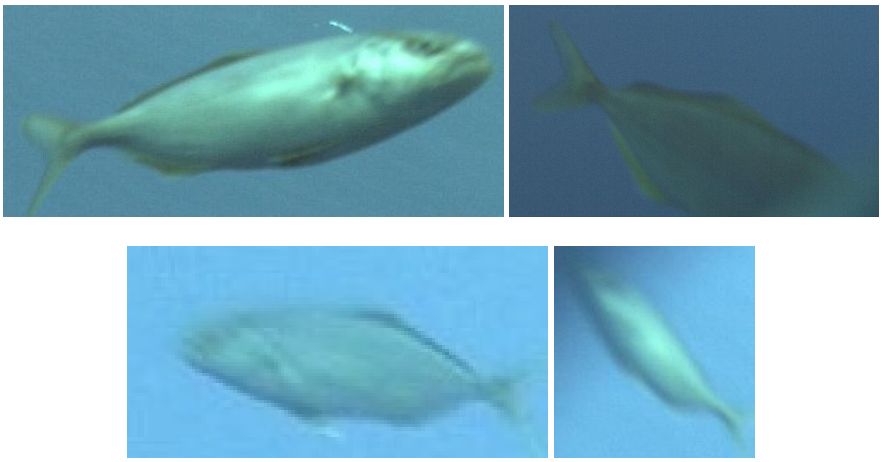


(**a**) Image from the Deep Themo deployment (**b**) *C. hippurus* segment

Figure 9. (**a**) An image captured in the Deep Themo deployment with one *Coryphaena hippurus*. (**b**) Segment of the fish extracted from the image.



(**a**) Examples of images from the Shallow Themo deployment



(**b**) *Seriola dumerili* segments

Figure 10. (**a**) Image captured in the Shallow Themo deployment. (**b**) Segments of *Seriola dumerili* extracted from the images.

Table 9. Accuracy of the multi-class classification of the fish segments obtained in the Deep Themo (24 images) and Shallow Themo (115 images) sea deployments. Each value is the average accuracy with 3 executions of training and classification. The training and validation has been done with the dataset presented in Table 2, in which the test data is integrated as part of the training data.

Architecture	Deep Themo	Shallow Themo
$4(C_{16}M)D_8D_{16}D_6$	47.22%	10.72%
$4(C_{16}M)D_{32}D_{32}D_6$	65.28%	26.96%
$5(C_{16}M)D_8D_6$	55.56%	33.91%
$5(C_{16}M)D_{32}D_6$	58.33%	37.10%
$5(C_{16}M)D_{16}D_8D_6$	65.28%	34.49%
VGG16	86.11%	32.46%
Chollet	52.78%	31.30%

5. Conclusions

Underwater fish recognition via autonomous probes that rely on constrained embedded computers is a challenging tasks, due to limitations on computational power, memory availability, and image quality, among others. Considering the above constraints, in this work we engineered convolutional neural network models to yield architectures of lim-

ited complexity and much fewer trainable parameters than complex object recognition architectures such as VGG16 or MobileNet. We showed that our solutions produce results comparable to such much larger and broadly used architectures, especially when aided by a computationally inexpensive object tracking algorithm that combines multiple classified images to yield a more accurate output.

Our experience in this work suggests that the most challenging task is the acquisition and preparation of a dataset. In general, the number of relevant images available for the classes of interest is limited, especially considering that the images should be captured under water. We compensated for the lack of images by extracting relevant segments from video frames, and mitigated the higher dataset similarity that results by placing nearly identical images only in one subset among training, validation, and test.

The imbalance of the training data also posed a significant challenge. As seen in Table 2, the number of images collected for the different species of interest in our work is very different. We argue that this is one possible reason causing a lower classification accuracy in our models: for binary classification, where the six classes have been split into two categories by maintaining a similar number of images per class, some of our architectures yield comparable results with respect to VGG16, which has two orders of magnitude more parameters. Future work includes collecting more images to balance the dataset, as this will result in a performance improvement for our models.

Author Contributions: Conceptualization, R.P. and A.F.A.; methodology, M.P., R.P. and A.F.A.; software, M.P., R.P. and A.F.A.; validation, M.P., P.C., E.B. and A.F.A.; experiments and data collection, R.P., P.C., E.B., A.S. and D.T.; writing and revision, all authors. All authors have read and agreed to the published version of the manuscript.

Funding: This work has received support from the European Union's Horizon 2020 Research and Innovation Programme under grant agreement no. 773753 (SYMBIOSIS), and from the Italian Ministry for University and Research (MIUR) under the initiative "Departments of Excellence" (Law 232/2016). We also gratefully acknowledge the support of NVIDIA Corporation for the donation of the Jetson TX-2 board used to develop the real-time version of our algorithm.

Acknowledgments: The authors would like to thank Deborah Levy and Tali Treibitz for sharing their tool to segment image sections that contain fish specimens [28]. The authors are also thankful to the University of Haifa team and collaborating personnel for carrying out the sea trials under the coordination of Roee Diamant.

Conflicts of Interest: The authors declare no conflict of interest.

Notes

1 http://symbiosis.networks.imdea.org/, accessed: May 26, 2022.
2 https://www.kaggle.com/datasets/sripaadsrinivasan/fish-species-image-data, accessed: May 26, 2022.
3 Labeled Fishes in the Wild, https://swfscdata.nmfs.noaa.gov/labeled-fishes-in-the-wild/, accessed: May 26, 2022.
4 Texas A&M—University of Haifa Eastern Mediterranean Observatory, http://themo.haifa.ac.il/

References

1. Yang, X.; Zhang, S.; Liu, J.; Gao, Q.; Dong, S.; Zhou, C. Deep learning for smart fish farming: Applications, opportunities and challenges. *Rev. Aquac.* **2021**, *13*, 66–90. https://doi.org/https://doi.org/10.1111/raq.12464.
2. Bojarski, M.; Yeres, P.; Choromanska, A.; Choromanski, K.; Firner, B.; Jackel, L.D.; Muller, U. Explaining How a Deep Neural Network Trained with End-to-End Learning Steers a Car. *arXiv* **2017**, arXiv:abs/1704.07911.
3. Shaikhina, T.; Khovanova, N.A. Handling limited datasets with neural networks in medical applications: A small-data approach. *Artif. Intell. Med.* **2017**, *75*, 51–63.
4. Patel, V.L.; Shortliffe, E.H.; Stefanelli, M.; Szolovits, P.; Berthold, M.R.; Bellazzi, R.; Abu-Hanna, A. The coming of age of artificial intelligence in medicine. *Artif. Intell. Med.* **2009**, *46*, 5–17.
5. Kononenko, I. Machine learning for medical diagnosis: History, state of the art and perspective. *Artif. Intell. Med.* **2001**, *23 1*, 89–109.

6. van Sloun, R.J.G.; Demi, L. Localizing B-Lines in Lung Ultrasonography by Weakly Supervised Deep Learning, In-Vivo Results. *IEEE J. Biomed. Health Inform.* **2019**, *24*, 957–964. https://doi.org/10.1109/JBHI.2019.2936151.
7. Berriel, R.F.; Lopes, A.T.; Rodrigues, A.; Varejão, F.M.; Oliveira-Santos, T. Monthly energy consumption forecast: A deep learning approach. In Proceedings of the International Joint Conference on Neural Networks (IJCNN), Anchorage, AK, USA, 14–19 May 2017; pp. 4283–4290.
8. Dias, P.A.; Tabb, A.; Medeiros, H. Apple flower detection using deep convolutional networks. *arXiv* **2018**, arXiv:abs/1809.06357.
9. Singh, S.P.; Kumar, A.; Darbari, H.; Singh, L.; Rastogi, A.; Jain, S. Machine translation using deep learning: An overview. In Proceedings of the 2017 International Conference on Computer, Communications and Electronics (Comptelix), Jaipur, India, 1–2 July 2017; pp. 162–167.
10. Amodei, D.; Ananthanarayanan, S.; Anubhai, R.; Bai, J.; Battenberg, E.; Case, C.; Casper, J.; Catanzaro, B.; Chen, J.; Chrzanowski, M.; et al. Deep Speech 2 : End-to-End Speech Recognition in English and Mandarin. *arXiv* **2016**, arXiv:abs/1512.02595.
11. van den Oord, A.; Dieleman, S.; Zen, H.; Simonyan, K.; Vinyals, O.; Graves, A.; Kalchbrenner, N.; Senior, A.W.; Kavukcuoglu, K. WaveNet: A Generative Model for Raw Audio. *arXiv* **2016**, arXiv:abs/1609.03499.
12. Zhao, Z.Q.; Zheng, P.; tao Xu, S.; Wu, X. Object Detection With Deep Learning: A Review. *IEEE Trans. Neural Networks Learn. Syst.* **2019**, *30*, 3212–3232.
13. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. https://doi.org/10.1038/nature14539.
14. Vargas, R.; Mosavi, A.; Ruiz, R. Deep learning: A review. In *Advances in Intelligent Systems and Computing*; Springer, Switzerland, 2017.
15. Diamant, R.; Voronin, V.; Kebkal, K.G. Design Structure of SYMBIOSIS: An Opto-Acoustic System for Monitoring Pelagic Fish. In Proceedings of the MTS/IEEE OCEANS, Marseille, France, 17–20 June 2019; pp. 1–6. https://doi.org/10.1109/OCEANSE.2019.8867440.
16. Rawat, W.; Wang, Z. Deep Convolutional Neural Networks for Image Classification: A Comprehensive Review. *Neural Comput.* **2017**, *29*, 2352–2449.
17. Iqbal, M.A.; Wang, Z.; Ali, Z.A.; Riaz, S. Automatic Fish Species Classification Using Deep Convolutional Neural Networks. *Wirel. Pers. Commun.* **2021**, *116*, 1043–1053. https://doi.org/10.1007/s11277-019-06634-1.
18. Cui, S.; Zhou, Y.; Wang, Y.; Zhai, L. Fish Detection Using Deep Learning. *Appl. Comput. Intell. Soft Comput.* **2020**, *2020*, 3738108. https://doi.org/10.1155/2020/3738108.
19. FishVerify Application. Available online: https://www.fishverify.com/ (accessed on 17 February 2021).
20. Technologies, R. FishFace Application. Available online: https://www.refind.se/blog/2018/11/15/fish-face-successful-installation-on-board (accessed on 17 February 2021).
21. Ravanbakhsh, M.; Shortis, M.R.; Shafait, F.; Mian, A.S.; Harvey, E.S.; Seager, J.W. Automated Fish Detection in Underwater Images Using Shape-Based Level Sets. *Photogramm. Rec.* **2015**, *30*, 46–62.
22. Li, X.; Shang, M.; Qin, H.; Chen, L. Fast accurate fish detection and recognition of underwater images with Fast R-CNN. In Proceedings of the MTS/IEEE OCEANS, Washington, DC, USA, 19–22 October 2015; pp. 1–5.
23. Villon, S.; Chaumont, M.; Subsol, G.; Villéger, S.; Claverie, T.; Mouillot, D. Coral Reef Fish Detection and Recognition in Underwater Videos by Supervised Machine Learning: Comparison Between Deep Learning and HOG+SVM Methods. In Proceedings of the ACIVS, Lecce, Italy, 24–27 October 2016.
24. Fish4Knowledge Consortium. Project Overview. Available online: https://groups.inf.ed.ac.uk/f4k/overview.htm (accessed on 17 February 2021).
25. Fishial.ai. Project Overview. Available online: https://fishial.ai/stories/2019/09/09/fishial-ai-project-overview/ (accessed on 17 February 2021).
26. Boudhane, M.; Nsiri, B. Underwater image processing method for fish localization and detection in submarine environment. *J. Vis. Commun. Image Represent.* **2016**, *39*, 226–238.
27. Lin, T.Y.; Goyal, P.; Girshick, R.B.; He, K.; Dollár, P. Focal Loss for Dense Object Detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**, *42*, 318–327.
28. Levy, D.; Belfer, Y.; Osherov, E.; Bigal, E.; Scheinin, A.P.; Nativ, H.; Tchernov, D.; Treibitz, T. Automated Analysis of Marine Video with Limited Data. In Proceedings of the IEEE/CVF CVPRW Workshop, Salt Lake City, UT, USA, 18–23 June 2018; pp. 1466–1468.
29. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv* **2015**, arXiv:abs/1409.1556.
30. Chollet, F. Building Powerful Image Classification Models Using Very Little Data. 2016. Available online: https://blog.keras.io/building-powerful-image-classification-models-using-very-little-data.html (accessed on 17 February 2021).
31. Picon, A.; Alvarez-Gila, A.; Seitz, M.; Ortiz-Barredo, A.; Echazarra, J.; Johannes, A. Deep convolutional neural networks for mobile capture device-based crop disease classification in the wild. *Comput. Electron. Agric.* **2019**, *161*, 280–290. https://doi.org/10.1016/j.compag.2018.04.002.
32. Howard, A.; Pang, R.; Adam, H.; Le, Q.; Sandler, M.; Chen, B.; Wang, W.; Chen, L.C.; Tan, M.; Chu, G.; et al. Searching for MobileNetV3. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea, 27–28 October 2019; pp. 1314–1324. https://doi.org/10.1109/ICCV.2019.00140.
33. Paraschiv, M.; Padrino, R.; Casari, P.; Fernández Anta, A. Very Small Neural Networks for Optical Classification of Fish Images and Videos. In Proceedings of the MTS/IEEE OCEANS, Singapore, 5–31 October 2020.

34. Bewley, A.; Ge, Z.; Ott, L.; Ramos, F.T.; Upcroft, B. Simple online and realtime tracking. In Proceedings of the 2016 IEEE International Conference on Image Processing (ICIP), Phoenix, AZ, USA, 25–28 September 2016; pp. 3464–3468.
35. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016.
36. Wojke, N.; Bewley, A.; Paulus, D. Simple online and realtime tracking with a deep association metric. In Proceedings of the 2017 IEEE International Conference on Image Processing (ICIP), Beijing, China, 17–20 September 2017; pp. 3645–3649.
37. Bewley, A. SORT Algorithm Source. Available online: https://github.com/6o6o/sort/ (accessed on 17 February 2021).
38. SMOT Challenge 2015 (Multi-Target Tracking). Available online: https://motchallenge.net/ (accessed on 17 February 2021).
39. Dubrovinskaya, E.; Casari, P. Underwater Direction of Arrival Estimation using Wideband Arrays of Opportunity. In Proceedings of the MTS/IEEE OCEANS, Marseille, France, 17–20 June 2019; pp. 1–7. https://doi.org/10.1109/OCEANSE.2019.8867262.
40. Dubrovinskaya, E.; Kebkal, V.; Kebkal, O.; Kebkal, K.; Casari, P. Underwater Localization via Wideband Direction-of-Arrival Estimation Using Acoustic Arrays of Arbitrary Shape. *Sensors* **2020**, *20*, 3862. https://doi.org/10.3390/s20143862.