sustainability

MDPI

Article

A Scalable Open-Source Framework for Machine Learning-Based Image Collection, Annotation and Classification: A Case Study for Automatic Fish Species Identification

Catarina N. S. Silva [1,*], Justas Dainys [1], Sean Simmons [2], Vincentas Vienožinskis [3] and Asta Audzijonyte [1,4]

[1] Nature Research Centre, LT-08412 Vilnius, Lithuania
[2] MyCatch and Angler's Atlas, Prince George, BC V2L 4S1, Canada
[3] Deeper, LT-10312 Vilnius, Lithuania
[4] Institute for Marine and Antarctic Studies, University of Tasmania, Hobart 7004, Australia
* Correspondence: catari.bio@gmail.com

Abstract: Citizen science platforms, social media and smart phone applications enable the collection of large amounts of georeferenced images. This provides a huge opportunity in biodiversity and ecological research, but also creates challenges for efficient data handling and processing. Recreational and small-scale fisheries is one of the fields that could be revolutionised by efficient, widely accessible and machine learning-based processing of georeferenced images. Most non-commercial inland and coastal fisheries are considered data poor and are rarely assessed, yet they provide multiple societal benefits and can have substantial ecological impacts. Given that large quantities of georeferenced fish images are being collected by fishers every day, artificial intelligence (AI) and computer vision applications offer a great opportunity to automate their analyses by providing species identification, and potentially also fish size estimation. This would deliver data needed for fisheries management and fisher engagement. To date, however, many AI image analysis applications in fisheries are focused on the commercial sector, limited to specific species or settings, and are not publicly available. In addition, using AI and computer vision tools often requires a strong background in programming. In this study, we aim to facilitate broader use of computer vision tools in fisheries and ecological research by compiling an open-source user friendly and modular framework for large-scale image storage, handling, annotation and automatic classification, using cost- and labour-efficient methodologies. The tool is based on TensorFlow Lite Model Maker library, and includes data augmentation and transfer learning techniques applied to different convolutional neural network models. We demonstrate the potential application of this framework using a small example dataset of fish images taken through a recreational fishing smartphone application. The framework presented here can be used to develop region-specific species identification models, which could potentially be combined into a larger hierarchical model.

Keywords: recreational fisheries; artisanal fisheries; citizen science; deep learning; fish species identification; image annotation; smart phone applications

Citation: Silva, C.N.S.; Dainys, J.; Simmons, S.; Vienožinskis, V.; Audzijonyte, A. A Scalable Open-Source Framework for Machine Learning-Based Image Collection, Annotation and Classification: A Case Study for Automatic Fish Species Identification. *Sustainability* **2022**, *14*, 14324. https://doi.org/10.3390/su142114324

Academic Editors: Mafalda Rangel, Hugo Diogo and Pablo Pita

Received: 27 June 2022
Accepted: 25 October 2022
Published: 2 November 2022

1. Introduction

More than 80% of global catches occur in fisheries that lack essential data, resources and infrastructure for stock assessments to be performed [1]. This is especially true for recreational fisheries, which continue to grow in popularity and provide important well-being and economic benefits, but remain hard to monitor, control and assess [2]. Given the large number of people engaged in recreational fisheries and the generally high level of technology used [3,4], there is a potential for large-scale data collection that could greatly improve our knowledge about recreational catches and fish population status.

Recreational fishers are typically motivated to conserve fish stocks, and experience with other groups has shown that engagement in citizen science programs both helps to generate large datasets and also promotes awareness and sense of stewardship [5]. For recreational fisheries management, citizen science would be especially powerful as it could enable more collaborative research and management [6,7].

Artificial intelligence (AI) has already revolutionised weather forecasting [8], wildfire disaster response [9], healthcare [10] and transportation [11]. AI and computer vision applications also offer an untapped opportunity to transform recreational fisheries management. This is because AI allows the rapid processing of angler contributed images, providing automatic species identification, and potentially also fish body size information. Even though research applying AI in fisheries has been increasing, with about 40 scientific publications per year [12], this is still very limited when compared to other fields. Currently, AI-based models are mainly applied to commercial fisheries [13,14], often targeting only a few species in a specific setting. Moreover, the methods, tools and scripts developed in these studies are often not publicly available, limiting their wider uptake, application and community-driven improvement. Finally, the AI application and development of new models still requires a solid background in programming and advanced computing. Yet, as new libraries and tools are being developed, the programming aspect can be increasingly streamlined, which would enable smaller research groups around the world to use these tools for their purposes. The most labour- and time-consuming part of developing species identification models is the collection of expert identified images, and not computational resources. Here, community contributions would be especially powerful, as research groups around the world could engage local citizen scientists to assemble images and develop species identification models relevant for local fisheries' management and ecological questions. Such models could then enable faster processing of new local citizen science data, spearheading marine recreational fishing assessments into a new data-rich era.

In this study, we aim to facilitate the use of AI and computer vision tools for recreational and small-scale fisheries research and management by presenting a scalable user-friendly AI framework to develop fish species identification models. We use a range of existing libraries and tools and add our own scripts and methods to combine them into one workflow. In Section 2 (Framework), we introduce this framework for analysing large image collections, which is accompanied by accessible scripts and jargon-free descriptions. The framework includes steps for: (1) data pre-processing; (2) image processing; and (3) machine learning model development. Alongside the framework, we also summarise currently available open-source image annotation tools and provide Jupyter notebooks with scripts that can be customised by researchers and applied to different types of imagery data. In Section 3 (Pilot case-study), we demonstrate the implementation of the framework and its potential use for recreational fisheries research, using a small example case study that aims to automate detection of fish species from images uploaded to a smartphone fishing application. Finally, in Section 4 (Lessons learned, challenges and future applications) we summarise the knowledge gained from the case study application and outline the main challenges and potential future development.

2. Framework

The framework developed in this study is summarised in Figure 1, and is divided into three main modules: data pre-processing; image processing; and machine learning. The different components and scripts can be customized to multiple image classification-oriented problems, such as species or individual identification, size estimation or identification of other phenotypic or morphological patterns.

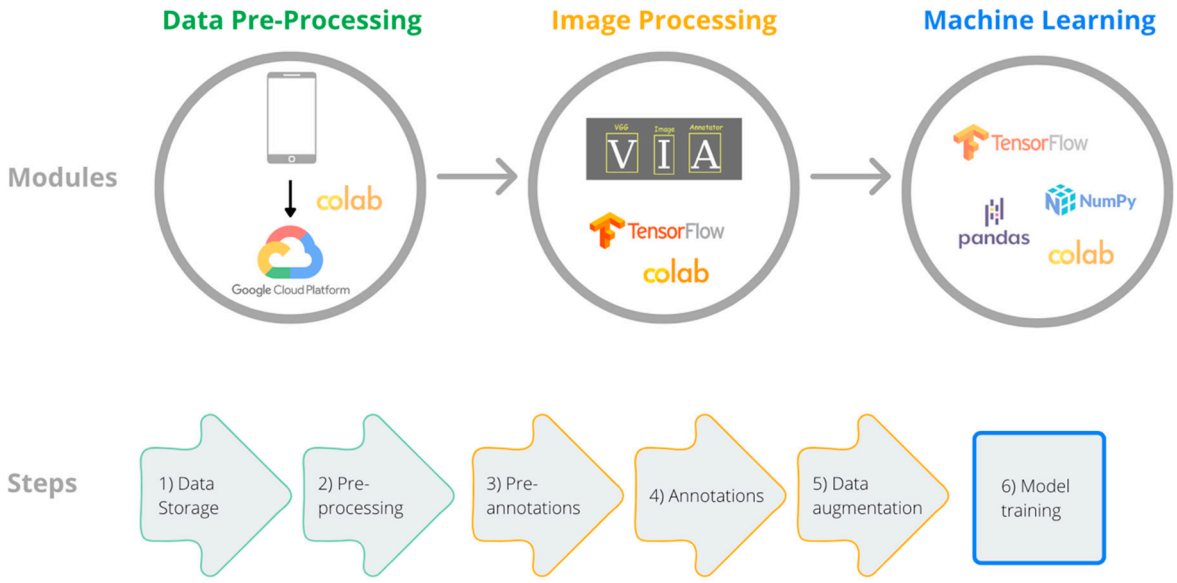

Figure 1. Overview of the framework, with the main tools used, consisting of three modules and six steps. Platforms and specific tools (e.g., Google Colab and Google Cloud Platform) indicated here were used in the example application, but could be replaced with other tools, as explained below.

Computer vision is a branch of computer science which aims to extract information from images [15] and develop visual recognition systems. Some of the most-used computer vision methods are image classification and object detection. Image classification is a technique used to classify a single object in an image or simply classify the entire image into user-defined categories (i.e., single-label classification; [16]). This method could be used in cases where there is only one fish in an image. Object detection (sometimes called object classification) is used to first detect the location of one or more objects in an image, and then classify each of the detected objects; it would be useful in cases where images contain multiple fish individuals of different species. Images used to train object detection algorithms must have manual annotations of the objects that should be detected, such as a rectangle—also known as a bounding box—placed around the object (for example, see [17]).

2.1. Module 1: Data Pre-Processing

The framework presented here assumes that users already have acquired identified images. These images might have been provided by citizen science programs, social media scans or targeted image collection. The model quality will depend on the accuracy of species identification in the training datasets, so expert knowledge in this step is important.

Step 1: Storage

In the framework proposed here, the images and associated metadata are stored on remote servers, i.e., cloud storage. Local storage could also be used, but there are multiple benefits to cloud storage of large datasets, including costs, facilitated collaboration, easy access from multiple devices, access to virtual servers used for analyses, efficient back-up, centralisation and data protection (see Supplementary Table S1 for description and comparison of the three main cloud storage providers).

Step 2: Sensitive data protection

For all further steps in this study, we used Google Colab [18], an online tool with built-in interactive Python programming environments such as Jupyter notebooks [19] and with access to hours of free computing resources such as graphics processing units (GPUs). If preferable, Google Colab could be replaced with other tools such as JupyterLab or Kaggle.

Images collected by citizens or extracted from the internet may contain sensitive data, such as people's faces. Depending on the nature of subsequent work (e.g., crowd-sourced annotation of images), it may be preferable to remove such data. In this framework, we introduce a step that uses face detection algorithms to remove sensitive data before further analysis (Figure 2). This is done by using a publicly available script *face_detection_overlay.ipynb*. This script uses the function *detect_face()* of the Python library CVlib and the pre-trained model *caffemodel* to detect human faces [20]. After face detection, a rectangle is drawn around the face, and an overlay is applied to the image content inside the rectangle (Figure 2).

Figure 2. Example of visual data before and after the face detection algorithm. Public domain image source: https://unsplash.com/photos/xp3xtQW3pqs (accessed on 5 September 2022).

2.2. Module 2: Image Processing

Before we apply computer vision methods, available images must be properly prepared. This includes classification of images into correct groups (i.e., fish species), placing boxes or tracing focus objects in the images and augmenting the dataset. Image processing is often done manually, and can be time consuming. This is particularly true if images for model training are collected through recreational fishing apps, citizen science or automated 'scrubbing' of online sites, as such image collections may include multiple irrelevant images. Below we present a few steps that can partly automate and speed up image processing tasks. Note that in our framework we focus on image classification and object detection using bounding boxes, and do not include other computer vision tools such as image

segmentation, where model training requires annotations with detailed tracing of object shapes. This is because the bounding box method is least time- and therefore resource-intensive, which was an important criterion given the focus of our study on developing tools for research groups with limited resources.

Step 3: Pre-annotations—accelerate manual annotation of images

Although expert-based manual annotation of images (into correct species or other groups that will be used in the model) cannot be avoided, there are several pre-annotation steps that can reduce the amount of manual work required. Our framework includes scripts for importing images from cloud storage and running an object detector using the module *inception_resnet_v2*—Keras image classification model pre-trained on Open Images Dataset V4 [21]. This object detector model was used to automatically place bounding boxes around the fish shapes in the image. In theory, the bounding box step is only needed for object detection, and not for image classification algorithms. If users are only interested in classification, the bounding box step can be skipped. However, automatic detection of shapes can still be useful in object detection applications, when dealing with image collections from diverse contributors and where some images may not even have the object that needs to be classified. For example, photos collected through angler apps may include pictures of location, gear or just accidental images. Running an object detection step will reduce the amount of work required to sort the images manually. For example, only images with the label "fish" in the annotations file, where a fish was detected by the object detector model, will be selected for further processing. In our case, we used the pre-trained model to find fish shapes, but it can be used to detect 600 other shapes, including elephant, lynx, bird, insect, shellfish, tree, plant and others. The final part of step 3 is converting the metadata of bounding boxes to absolute coordinates and saving them in a VGG format (in a .csv file), as this format is required for subsequent manual annotation (VGG software, see below). This last section of the script can be easily customised for other annotation tool formats. The scripts for step 3 are available in the notebook *object_detection_pre-annotation.ipynb*.

Step 4: Annotations—manual annotation of images

Once the available image collection has been pre-annotated, the next step requires expert-based classification or annotation of images. There is a variety of open-source software tools for manual image annotation (Supplementary Table S2) with a range of formats for importing and exporting object annotations; each software package typically has its own Json- or csv-based format. Some tools are only available online, which means that images must be uploaded to annotation servers. This may limit their application for sensitive data, or in situations where experts engaged in image annotation have limited internet connectivity.

In our example case study, we used the VGG software [22], as it could be run locally and had easy setup and installation. The .csv file generated in Step 3 (*object_detection_pre-annotation.ipynb*) was opened in the VGG software, where automatic pre-annotation of image shapes was inspected manually and corrected if needed (with bounding boxes adjusted to better fit the object), and class names were added. In our case, class names included identification of fish species, which required expert knowledge. If class names are provided automatically from the pre-annotation step (e.g., the model only aims to classify fishes or other shapes), they can also be corrected manually at this stage.

TensorFlow Lite Model Maker requires annotation input files in a specific format, so we thus provide a script (*convert_annotations_VGG_to_TF.ipynb*) to format the .csv file from VGG software to the TensorFlow format. This includes converting bounding box coordinates from absolute values generated by the VGG software to relative values needed for TensorFlow, and splitting the dataset into train, test and validation sets.

Step 5: Data augmentation

Data augmentation involves creating multiple copies of the same images, but with transformations such as flipping, rotating, scaling and cropping. Data augmentations

can help reduce overfitting in deep convolutional neural networks [23], improve performance [23,24], model convergence [25], generalisation and robustness on out-of-distribution samples [26,27]. Depending on the method of computer vision used, data augmentation steps will differ. For the image classification method, data augmentation only requires transforming (rotating, flipping) the images and saving them as new images. For the object detection method, where objects have metadata on bounding boxes, image transformation steps must also be applied to bounding boxes, i.e., the coordinates of the bounding boxes must be adjusted accordingly, depending on the rotations or flips of the images.

In this framework, we use the open source Albumentations library [28] for data augmentation. The script *data_augmentation_classification.ipynb* defines an augmentation pipeline for the image classification approach. In this script, we first use the function *A.Compose()* from the Albumentations library to define image transformations (vertical and horizontal flips), and then loop over all images in a directory to apply these transformations and save the new images. In the script *data_augmentation_object_detection.ipynb*, we then apply these transformations (vertical and horizontal flips) to the annotations (bounding boxes) file. At the end of the script, the transformed coordinates of the bounding boxes are saved as a new annotation file in a .csv format. The file is then used as an input to train an object detection model.

2.3. Module 3: Machine Learning

Step 6: Model training and testing

The framework presented here uses the Tensorflow Lite Model Maker library [29,30] and transfer learning, which reduces the amount of training data required and model training time when compared to a model trained from scratch. Transfer learning is a technique where the knowledge is transformed from one model to another. For image classification tasks, pre-trained models have been typically trained on the ImageNet dataset [31,32]. In most cases, the categories identified in this pre-trained model are quite broad—i.e., in our case, the most suitable category was "fish". Tensorflow Lite supports several pre-trained models, including EfficientNet-Lite, MobileNetV2 and ResNet50 [33–35], which are models for image classification, and EfficientDet-Lite [0–4], a family of models for object detection derived from the EfficientDet model [36]. In the first step of the transfer learning approach, one pre-trained model is selected, and the parameters of the model (as classification weights and layers) are used to generate the most useful features of the new images. Therefore, it is important to select a model pre-trained for a similar classification task that can be identified by testing more than one pre-trained model. As ImageNet includes classes such as "fish", we hypothesised that the pre-trained models used in Tensorflow Lite Model Maker Library would perform well for fish species classification. Finally, the library is flexible, and new pre-trained models can be added by customising the library code. The Lite version of the library also means that it runs relatively fast, and could even be integrated into mobile phone applications.

For this framework, we developed the script *image_classification.ipynb* to train and test (evaluate) an image classification model using the pre-trained models mentioned above. This means that we "update" the pre-trained model with new categories of higher resolution (specific fish species). This script also generates a confusion matrix for visualising model performance, and functions to load a newly trained model and run classification inference on new images. A script *object_detection.ipynb* goes through similar steps of training and testing models, but is used to train and test an object detection model.

3. Pilot Case-Study

To illustrate the feasibility of the framework developed here, we present a pilot case-study to develop the automatic identification of six fish species—Common bream (*Abramis brama*); European carp (*Cyprinus carpio*); Northern pike (*Esox lucius*); Largemouth bass (*Micropterus salmoides*); European perch (*Perca fluviatilis*); and Pikeperch (*Sander lucioperca*) (Figure 3; Table 1)—using images from a recreational fishing smart phone

application. Images for model training and testing were obtained through a collaborative agreement with a company Fish Deeper™, which provides fishfinder sonar devices and an associated smart phone application where anglers can log their catch. The anonymous data obtained included images and associated metadata, such as GPS coordinates, and where available, fish species identification by the user, fish length, fish weight, fishing technique and bait used. After the automated pre-annotation to select only images with fish, the manual image annotation was done by an experienced researcher and fish biologist (Justas Dainys) with the required expertise in fish species identification (see discussion at the end for more details about the time used in this step). Next, we applied image augmentation (vertical and horizontal flips) to increase the number of images for model training, which provided two additional images for each original photo, and resulted in a total of 4809 images.

Figure 3. Example of fish images used for model training and testing.

Table 1. Sample sizes (number of annotated images) used for the image classification and object detection models in the example case study.

Species	Common Name	Number of Annotated Images	Total Number of Images after Augmentation
Abramis brama	Common bream	111	333
Cyprinus carpio	European carp	377	1131
Esox lucius	Northern pike	420	1260
Micropterus salmoides	Largemouth bass	175	525
Perca fluviatilis	European perch	332	996
Sander lucioperca	Pikeperch	188	564

Choosing the best parameters such as batch size or number of epochs for a learning algorithm is an important step when tuning a machine learning algorithm. Batch size defines the number of samples used in one training step. For example, if it is set to 1, the

model weights are updated after each training example. This can result in faster learning, but also adds instability as the weights vary widely. The number of epochs corresponds to the number of times that the learning algorithm will work through the entire training dataset. More epochs could achieve better accuracy, though training for too many epochs may lead to overfitting. The choice of the best model parameters depends on factors such as dataset size, number of classes and similarity between classes. However, model fine tuning is beyond the objectives of this study, and we have conducted only a small exploration of model parameters.

The best performance for the image classification was achieved when using EfficientNet-Lite0 model architecture, a batch size of 32 and 20 epochs, with an overall accuracy of 0.91 and mean loss of 0.71 (Figure 4). Many classes had high precision values, although the precision for *Sander lucioperca* was quite low. From the confusion matrix (Figure 4), *Sander lucioperca* were commonly mistaken for *Esox lucius*, *Abramis brama* and *Perca fluviatilis*. From the remaining models tested, overall accuracy varied between 0.53 (ResNet50, batch size of 8 and 50 epochs) and 0.81 (EfficientNet-Lite, batch size of 32 and 20 epochs), while mean loss varied between 5.31 (ResNet50, batch size of 8 and 50 epochs) and 0.89 (EfficientNet-Lite, batch size of 32 and 20 epochs).

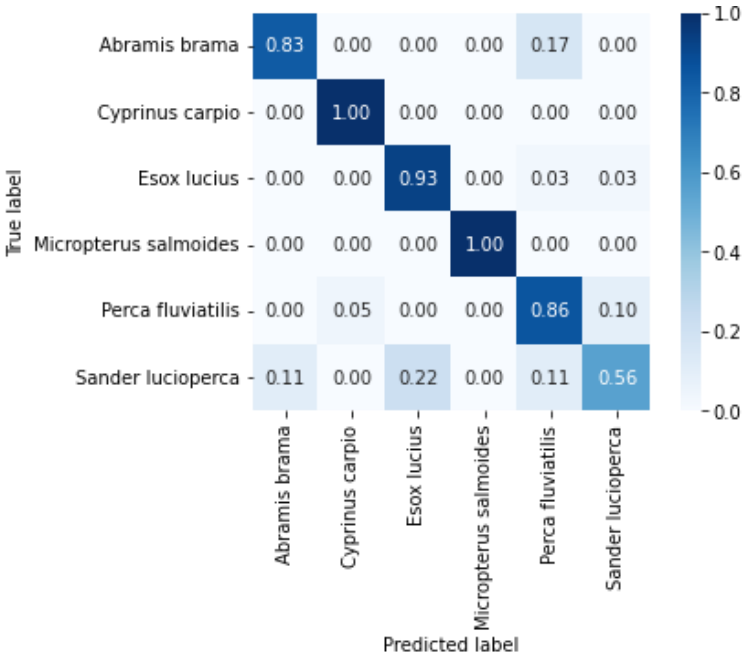

Figure 4. Confusion matrix of the image classification results with normalised, relative values of correct predictions for each species (i.e., precision) obtained when using EfficientNet-Lite0 model architecture, with a batch size of 32 and 20 epochs.

For object detection, the best overall precision obtained was 0.48 when using EfficientDet-Lite0 model architecture, with a batch size of 32 and 20 epochs (Table 2). From the remaining models tested, the overall precision varied between 0.41 (EfficientDet-Lite0, batch size of 2, 50 epochs) and 0.47 (EfficientDet-Lite0, batch size of 8, 50 epochs). Similar to what other researchers found (e.g., [37,38]), our results show that the classification method achieved higher overall performance compared to the object detection method. Training the model with a larger number of images, a balanced dataset and optimising hyperparameters (such as epochs and batch size) may improve model performance.

Table 2. Precision values for the best performing object detection model (model architecture = EfficientDet-Lite0, batch size = 32, epochs = 20).

Species	Common Name	Precision
All (Average Precision)		0.48
Abramis brama	Common bream	0.20
Cyprinus carpio	European carp	0.75
Esox lucius	Northern pike	0.55
Micropterus salmoides	Largemouth bass	0.53
Perca fluviatilis	European perch	0.31
Sander lucioperca	Pikeperch	0.53

4. Potential Framework Applications and Challenges

In the example case study, we illustrate the scientific application, utility and potential for scalability of the AI framework for fish species or other object classification. The framework is flexible, and can be customised for a variety of image classification and research questions. We show that with a relatively small number of images per class (200–300), a model of reasonably high performance can be developed quickly and with minimal resources for a small number of classification classes (six in our case). Traditional deep learning-based approaches require a lot of computational power to train and then apply image classification models. To reduce the computation power, as well as the amount of data needed, we use the Tensorflow Lite Model Maker library [29,30], which uses transfer learning from a pre-trained model. The good model performance achieved in this pilot study using a small amount of data supports the effective transfer of useful information from the pre-trained model.

While applying the framework to our pilot case study, we required three months of work, and have used a total of 59.39 GiB of cloud storage at a total cost of €12.81. The computational resources used were 16 vCPUs, 60 GB RAM and a Nvidia Tesla P4 GPU, at a total cost of €117.20. The costs were also kept low because we took advantage of free online tools with Python programming environments and free computing resources (Google Colab) for most of the exploratory work. Paid services of computer engine resources and notebooks were used only for intensive model runs.

We found that in steps 2 and 3 the pre-trained models for detecting human faces or fish shapes were not always accurate, and could still be improved. The face detection model *caffemodel* sometimes failed to detect a face in a horizontal position (when images were taken in landscape mode). False positives also occurred when the model placed bounding boxes on fish "faces". This step could be improved by training the model to detect human faces using augmented data with transformations such as image rotation or vertical flips. Therefore, for images that need to be crowdsourced to public domains, e.g., manual annotations or citizen science projects, the potential for sensitive data leakage must be carefully addressed. When it comes to detecting fish shapes and placing bounding boxes around them, in most cases the pre-trained model *inception_resnet_v2* worked well, although often the box excluded small parts of the fish (usually the end of the tail). However, when there were many overlapping fish in the image, the model did not always detect all fish in an image. In a few images, the pre-trained model identified other objects (e.g., shoes, boxes, etc.) as fish.

Manual annotation of images—especially if it requires expert knowledge—is one of the bottlenecks for AI model development. The pre-annotation step (step 3) accelerated the process by automatically adding bounding boxes around fish, but the annotated images still had to be individually identified. On average, for the six common and clearly distinct species, annotating one photo took about 10–15 s. The data available from the angler app included photos from multiple fish species, but we only selected the six most common species for this example application. All available photos were divided into separate folders,

depending on the month they were taken. Each folder contained about 2000 photos, and their review and annotation took approximately three hours of concentrated work by a skilled expert. In theory, once the species identification model is developed, new photos can be identified faster by first applying the model, and then manually processing only those photos that had a low classification score, which therefore likely contain other species (the actual score would have to be tested).

Citizen scientists can also contribute to the manual annotation of images for AI projects. For example, Gundelund et al. [39] show that citizen scientists can estimate fisheries metrics and identify species with results comparable to surveys by scientists. If images can be shared publicly, crowdsourcing citizen science platforms such as Zooniverse (https://www.zooniverse.org/ last accessed on 24 October 2022) could speed up the annotation, depending on the level of skill required. For example, Anton et al. [40] used the platform to engage many citizen scientists to efficiently and accurately annotate data from underwater footage to detect cold water corals. To ensure higher accuracy, the authors used repeated annotations, i.e., each video clip was annotated by eight citizen scientists and an agreement threshold of 80% was used.

Like other researchers (e.g., [13]), we found that image augmentation through rotation and flips improved the image classification model performance, with overall accuracy increasing by 10%. As the dataset used in the pilot study was small, and this difference in model accuracy can vary with dataset size, these results need to be interpreted with caution. The augmentation was easy, fast and straightforward, and we recommend using it in most image classification and object detection applications. In our pilot case study, the process of training one model using the classification technique took 6903 s (approximately 2 h), while using the object detection technique it took 3906 s (approximately 1 h) for the same number of images (n = 4809), batches (n = 32) and epochs (n = 20), but with a different pre-trained model (EfficientNet-Lite0 and EfficientDet-Lite0, respectively). It is important to note that more computer time is needed to explore different parameters and pre-trained models, and these times also vary significantly when different parameters are used. For the same amount of data and hyperparameter space (batch size and epochs), image classification can require more time to train a model compared to the object detection technique. To achieve higher performance, however, object detection methods might need larger amounts of data and more time for hyperparameters optimisation.

This study shows that a simple and reasonably performing fish species identification model can be developed from a relatively small number of images and minimal computational resources; this means that various research groups could develop models for their own focus species [13,14,41]. The exciting and important future step would be to combine these models into a global, hierarchical classification framework for automatic identification of fish species around the world. Even though groups might use different techniques (such as image classification, object detection and segmentation), individual and regional models could be combined into an ensemble model. Usually, an ensemble classification model comprises two steps: (1) generating classification results using multiple weak classifiers (i.e., models that perform better than random guessing); and (2) integrating multiple results into a consistency function to obtain the final result with voting schemes [42]. There are different methods of ensemble learning with their own advantages and disadvantages (reviewed in [42]), and this area of research is rapidly evolving. Ensemble learning has already been recognised to improve the performance of individual models, and building a new model by ensemble learning requires less time, data and computational resources than training a new model with all the data combined.

Global open-access AI models for fish species identification would have a range of applications for research and fisheries management. For example, within fisheries management, AI models can be applied in the automatic electric monitoring of catches and in the detection of illegal, unregulated and unreported fishing (e.g., [43,44]). Data collected by fishers could be used for understanding species range shifts, abundance and population status. Platforms such as iNaturalist and Fishbase.org could benefit from

models developed at regional levels. As open-source species identification, size estimation or sex determination models improve they can be integrated into multiple data collection platforms, further accelerating data collection and knowledge sharing.

Supplementary Materials: The following supporting information can be downloaded at: https://www.mdpi.com/article/10.3390/su142114324/s1, Table S1: A comparison of the main storage services providers Amazon Web Services (AWS), Google Cloud Platform (GCP) and Microsoft Azure Platform (Azure) with specific terminology used by each provider; Table S2: List of open-source software for image annotation with details about export formats.

Author Contributions: Conceptualization, C.N.S.S. and A.A.; Data curation, J.D. and V.V.; Formal analysis, C.N.S.S. and J.D.; Funding acquisition, A.A.; Methodology, C.N.S.S. and A.A.; Project administration, A.A.; Resources, S.S. and V.V.; Supervision, A.A.; Visualization, C.N.S.S.; Writing–original draft, C.N.S.S.; Writing–review & editing, C.N.S.S., J.D., S.S., V.V. and A.A. All authors have read and agreed to the published version of the manuscript.

Funding: This study has received funding from European Regional Development Fund (Project No. 01.2.2-LMT-K-718-02-0006) under grant agreement with the Research Council of Lithuania (LMTLT).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: All steps described in this publication and scripts used are supported by a freely available code library, deposited in github.com/FishSizeProject/ML-framework-for-image-processing Training materials based on this publication are also available in a free online course "Machine learning based image collection, annotation and classification" available through https://fishsizeproject.github.io/Course-MLforImageProcessing/ (both sites last accessed on 24 October 2022).

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

References

1. Costello, C.; Ovando, D.; Hilborn, R.; Gaines, S.D.; Deschenes, O.; Lester, S.E. Status and Solutions for the World's Unassessed Fisheries. *Science* **2020**, *338*, 517–521. [CrossRef] [PubMed]
2. Meirelles, K.; Freire, F.; Belhabib, D.; Espedido, J.C.; Hood, L.; Kleisner, K.M.; Lam, V.W.L.; Machado, M.L.; Mendonça, J.T.; Meeuwig, J.J.; et al. Estimating Global Catches of Marine Recreational Fisheries. *Front. Mar. Sci.* **2020**, *7*, 1–18. [CrossRef]
3. Gordoa, A.; Dedeu, A.L.; Boada, J. Recreational Fishing in Spain: First National Estimates of Fisher Population Size, Fishing Activity and Fisher Social Profile. *Fish. Res.* **2019**, *211*, 1–12. [CrossRef]
4. Hyder, K.; Maravelias, C.D.; Kraan, M.; Radford, Z.; Prellezo, R. Marine Recreational Fisheries-Current State and Future Opportunities. *ICES J. Mar. Sci.* **2020**, *77*, 2171–2180. [CrossRef]
5. Dickinson, J.L.; Zuckerberg, B.; Bonter, D.N. Citizen Science as an Ecological Research Tool: Challenges and Benefits. *Annu. Rev. Ecol. Evol. Syst.* **2010**, *41*, 149–172. [CrossRef]
6. Venturelli, P.A.; Hyder, K.; Skov, C. Angler Apps as a Source of Recreational Fisheries Data: Opportunities, Challenges and Proposed Standards. *Fish Fish.* **2017**, *18*, 578–595. [CrossRef]
7. Harris, D.; Johnston, D.; Yeoh, D. More for Less: Citizen Science Supporting the Management of Small-Scale Recreational Fisheries. *Reg. Stud. Mar. Sci.* **2021**, *48*, 102047. [CrossRef]
8. Dewitte, S.; Cornelis, J.P.; Müller, R.; Munteanu, A. Artificial Intelligence Revolutionises Weather Forecast, Climate Monitoring and Decadal Prediction. *Remote Sens.* **2021**, *13*, 3209. [CrossRef]
9. Jaafari, A.; Zenner, E.K.; Panahi, M.; Shahabi, H. Hybrid Artificial Intelligence Models Based on a Neuro-Fuzzy System and Metaheuristic Optimization Algorithms for Spatial Prediction of Wildfire Probability. *Agric. For. Meteorol.* **2019**, *266–267*, 198–207. [CrossRef]
10. Yu, K.H.; Beam, A.L.; Kohane, I.S. Artificial Intelligence in Healthcare. *Nat. Biomed. Eng.* **2018**, *2*, 719–731. [CrossRef]
11. Abduljabbar, R.; Dia, H.; Liyanage, S.; Bagloee, S.A. Applications of Artificial Intelligence in Transport: An Overview. *Sustainability* **2019**, *11*, 189. [CrossRef]
12. Ebrahimi, S.H.; Ossewaarde, M.; Need, A. Smart Fishery: A Systematic Review and Research Agenda for Sustainable Fisheries in the Age of Ai. *Sustainability* **2021**, *13*, 6037. [CrossRef]
13. Lekunberri, X.; Ruiz, J.; Quincoces, I.; Dornaika, F.; Arganda-Carreras, I.; Fernandes, J.A. Identification and Measurement of Tropical Tuna Species in Purse Seiner Catches Using Computer Vision and Deep Learning. *Ecol. Inform.* **2022**, *67*, 101495. [CrossRef]

14. Ovalle, J.C.; Vilas, C.; Antelo, L.T. On the Use of Deep Learning for Fish Species Recognition and Quantification on Board Fishing Vessels. *Mar. Policy* **2022**, *139*, 105015. [CrossRef]
15. Prince, S.J.D. *Computer Vision: Models, Learning and Inference*; Cambridge University Press: Cambridge, UK, 2012.
16. Mohri, M.; Rostamizadeh, A.; Talwalkar, A. *Foundations of Machine Learning*; MIT Press: Cambridge, MA, USA, 2012.
17. dos Santos, A.A.; Gonçalves, W.N. Improving Pantanal Fish Species Recognition through Taxonomic Ranks in Convolutional Neural Networks. *Ecol. Inform.* **2019**, *53*, 100977. [CrossRef]
18. Bisong, E. Building Machine Learning and Deep Learning Models on Google Cloud Platform. In *Building Machine Learning and Deep Learning Models on Google Cloud Platform*; Apress: Berkeley, CA, USA, 2019; pp. 59–64. ISBN 9781484244708.
19. Kluyver, T.; Ragan-Kelley, B.; Pérez, F.; Granger, B.; Bussonnier, M.; Frederic, J.; Kelley, K.; Hamrick, J.; Grout, J.; Corlay, S.; et al. Jupyter Notebooks—A Publishing Format for Reproducible Computational Workflows. In *Positioning and Power in Academic Publishing: Players, Agents and Agendas, Proceedings of the 20th International Conference on Electronic Publishing, Göttingen, Germany, 10 June 2016*; IOS Press: Amsterdam, The Netherlands, 2016; pp. 87–90. [CrossRef]
20. Ponnusamy, A. Cvlib—High Level Computer Vision Library for Python. In Proceedings of the ICCS 2021: 21st International Conference on Computational Science, Krakow, Poland, 16–18 June 2018.
21. Kuznetsova, A.; Rom, H.; Alldrin, N.; Uijlings, J.; Krasin, I.; Pont-Tuset, J.; Kamali, S.; Popov, S.; Malloci, M.; Kolesnikov, A.; et al. The Open Images Dataset V4: Unified Image Classification, Object Detection, and Visual Relationship Detection at Scale. *Int. J. Comput. Vis.* **2020**, *128*, 1956–1981. [CrossRef]
22. Dutta, A.; Zisserman, A. The VIA Annotation Software for Images, Audio and Video. In Proceedings of the 27th ACM International Conference on Multimedia, Nice, France, 21–25 October 2019; pp. 2276–2279. [CrossRef]
23. Shorten, C.; Khoshgoftaar, T.M. A Survey on Image Data Augmentation for Deep Learning. *J. Big Data* **2019**, *6*, 60. [CrossRef]
24. Mikołajczyk, A.; Grochowski, M. Data Augmentation for Improving Deep Learning in Image Classification Problem. In Proceedings of the 2018 International Interdisciplinary PhD Workshop, Swinoujscie, Poland, 9–12 May 2018; pp. 117–122.
25. Liu, S.; Papailiopoulos, D.; Achlioptas, D. Bad Global Minima Exist and SGD Can Reach Them. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 8543–8552.
26. Hendrycks, D.; Mu, N.; Cubuk, E.D.; Zoph, B.; Gilmer, J.; Lakshminarayanan, B. AugMix: A Simple Data Processing Method to Improve Robustness and Uncertainty. In Proceedings of the International Conference on Learning Representations (ICLR), Addis Ababa, Ethiopia, 26 April–1 May 2020; pp. 1–15.
27. Bengio, Y.; Bastien, F.; Bergeron, A.; Boulanger-Lewandowski, N.; Breuel, T.; Chherawala, Y.; Cisse, M.; Côté, M.; Erhan, D.; Eustache, J.; et al. Deep Learners Benefit More from Out-of-Distribution Examples. In Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, Fort Lauderdale, FL, USA, 11–13 April 2011; pp. 164–172.
28. Buslaev, A.; Iglovikov, V.I.; Khvedchenya, E.; Parinov, A.; Druzhinin, M.; Kalinin, A.A. Albumentations: Fast and Flexible Image Augmentations. *Information* **2020**, *11*, 125. [CrossRef]
29. Abadi, M.; Agarwal, A.; Barham, P.; Brevdo, E.; Chen, Z.; Citro, C.; Corrado, G.S.; Davis, A.; Dean, J.; Devin, M.; et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. *arXiv* **2016**, arXiv:1603.04467.
30. Abadi, M.; Barham, P.; Chen, J.; Chen, Z.; Davis, A.; Dean, J.; Devin, M.; Ghemawat, S.; Irving, G.; Isard, M.; et al. TensorFlow.js. In Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation, Savannah, GA, USA, 2–4 November 2016.
31. Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. ImageNet Large Scale Visual Recognition Challenge. *Int. J. Comput. Vis.* **2015**, *115*, 211–252. [CrossRef]
32. Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; Fei-Fei, L. ImageNet: A Large-Scale Hierarchical Image Database. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009.
33. Tan, M.; Le, Q.V. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In Proceedings of the International Conference on Machine Learning, Long Beach, CA, USA, 10–15 June 2019; pp. 10691–10700.
34. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.C. MobileNetV2: Inverted Residuals and Linear Bottlenecks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018; pp. 4510–4520. [CrossRef]
35. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; Volume 2016, pp. 770–778. [CrossRef]
36. Tan, M.; Pang, R.; Le, Q.V. EfficientDet: Scalable and Efficient Object Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 10778–10787. [CrossRef]
37. Horn, G. Van; Mac, O.; Shepard, A.; Adam, H.; Song, Y.; Cui, Y.; Sun, C.; Perona, P.; Belongie, S. The INaturalist Species Classification and Detection Dataset—Supplementary Material. *Comput. Vis. Found.* **2017**, *32*, 4–6.
38. Zheng, Y.Y.; Kong, J.L.; Jin, X.B.; Wang, X.Y.; Su, T.L.; Zuo, M. Cropdeep: The Crop Vision Dataset for Deep-Learning-Based Classification and Detection in Precision Agriculture. *Sensors* **2019**, *19*, 1058. [CrossRef]
39. Gundelund, C.; Venturelli, P.; Hartill, B.W.; Hyder, K.; Olesen, H.J.; Skov, C. Evaluation of a Citizen Science Platform for Collecting Fisheries Data from Coastal Sea Trout Anglers. *Can. J. Fish. Aquat. Sci.* **2021**, *78*, 1576–1585. [CrossRef]
40. Anton, V.; Germishuys, J.; Bergström, P.; Lindegarth, M.; Obst, M. An Open-Source, Citizen Science and Machine Learning Approach to Analyse Subsea Movies. *Biodivers. Data J.* **2021**, *9*, e60548. [CrossRef]

41. Palmer, M.; Álvarez-Ellacuría, A.; Moltó, V.; Catalán, I.A. Automatic, Operational, High-Resolution Monitoring of Fish Length and Catch Numbers from Landings Using Deep Learning. *Fish. Res.* **2022**, *246*, 106166. [CrossRef]
42. Dong, X.; Yu, Z.; Cao, W.; Shi, Y.; Ma, Q. A Survey on Ensemble Learning. *Front. Comput. Sci.* **2020**, *14*, 241–258. [CrossRef]
43. Bartholomew, D.C.; Mangel, J.C.; Alfaro-Shigueto, J.; Pingo, S.; Jimenez, A.; Godley, B.J. Remote Electronic Monitoring as a Potential Alternative to On-Board Observers in Small-Scale Fisheries. *Biol. Conserv.* **2018**, *219*, 35–45. [CrossRef]
44. Poisson, F.; Budan, P.; Coudray, S.; Gilman, E.; Kojima, T.; Musyl, M.; Takagi, T. New Technologies to Improve Bycatch Mitigation in Industrial Tuna Fisheries. *Fish Fish.* **2022**, *23*, 545–563. [CrossRef]