





Article

Novel Algorithm to Detect, Classify, and Count Mussel Larvae in Seawater Samples Using Computer Vision

Pedro Orgeira-Crespo ^{1,*} , Carlos Gabín-Sánchez ² , Fernando Aguado-Agelet ³  and Guillermo Rey-González ¹ 

¹ Aerospace Area, Department of Mechanical Engineering, Heat Engines and Machines, and Fluids, Aerospace Engineering School, University of Vigo, Campus Orense, 32004 Orense, Spain; guillermo.rey@uvigo.es

² Centro de Investigaciones Mariñas (CIMA), Consellería do Mar, Xunta de Galicia, Vilanova de Arousa, 36611 Pontevedra, Spain

³ Telecommunication Engineering School, University of Vigo, 36310 Vigo, Spain; faguado@uvigo.es

* Correspondence: porgeira@uvigo.es

Featured Application: Automatic detection, classification, and accounting of mussel larvae in their different stages of growth using computer vision.

Abstract: The European Union's mussel production industry is dependent on obtaining mussel larvae as seed for cultivation, a process traditionally monitored through labor-intensive manual sampling and microscopic counting prone to human error and time-consuming procedures. To address these challenges, our research presents a computer vision-based methodology for accurately identifying, classifying, and quantifying mussel larvae individuals across various developmental stages from microscopic images of water samples. Utilizing a neural network architecture derived from the YOLO method, our approach integrates convolutional, pooling, and fully connected layers to automate detection, classification, and accounting tasks. Through training with manually labeled samples and employing data augmentation techniques, we established a robust framework capable of processing diverse larval specimens effectively. Our research not only streamlines mussel larvae monitoring processes but also underscores the potential of computer vision techniques to enhance efficiency and accuracy in aquaculture industries.

Keywords: computer vision; neural network; mussel larvae



Citation: Orgeira-Crespo, P.; Gabín-Sánchez, C.; Aguado-Agelet, F.; Rey-González, G. Novel Algorithm to Detect, Classify, and Count Mussel Larvae in Seawater Samples Using Computer Vision. *Appl. Sci.* **2024**, *14*, 5113. <https://doi.org/10.3390/app14125113>

Academic Editor: Christos Bouras

Received: 15 May 2024

Revised: 3 June 2024

Accepted: 10 June 2024

Published: 12 June 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Spain produced about 7 in every 10 tons of the EU's farmed Mediterranean mussels (*Mytilus galloprovincialis*) in 2022, largely due to its rafts in the estuaries of northern Spain using the 'off bottom' method [1]. The Mediterranean mussel (*Mytilus galloprovincialis*) is the most common species in the EU and represents 61% of the total community mussel production in the EU-28. This species is cultivated in Mediterranean countries (Italy, Greece, France, Spain, Bulgaria, Croatia, Slovenia), as well as in Galicia (Atlantic coast of Spain) [1]. Spain produces more than 250,000 tons per year (except in 2013, when red tide or algae blooms caused a drop in production) [2]. It is the largest producer in the EU (with 41% of Community production), followed by Italy, France, the Netherlands, Denmark, and Germany, with 8–12% of Community [1].

The supply of mussel seed is critical for the development of industrial mussel cultivation [3]. Mussel farming in Galicia (NW Spain) requires 9000 Tm of mussel seed per year to support the current mussel production rate [4]. A total of 66% of mussel seed used in mussel cultivation is obtained by scraping directly from intertidal exposed rocky shores where mussel seed is attached, although gathering of mussel seed from artificial collector ropes has increased in recent years [4], principally due to its higher growth rate when cultivated on the raft [5]. Since Galician legislature restricts the number of ropes per raft (maximum 600 ropes of 12 m length during the larval settlement season), it is

crucial to increase the seed yield obtained on artificial collectors by the development of new designs [6].

To improve knowledge about the presence and behavior of larvae in the sea, the Spanish administration has a sample collection program underway at seven points distributed in five Galician estuaries. Through this program, the presence of 3 larval stages can be identified: an earlier one (D-shaped larvae, 3 to 4 days old, with an average size of $40\ \mu\text{m}$), a more advanced stage, close to metamorphosis and subsequent fixation (umbonate larvae, aged between 25 and 30 days and with an average size of $180\ \mu\text{m}$), and an intermediate one ($100\ \mu\text{m}$). This information is of great interest to the sector since it warns of the optimal moments for the placement of the collector ropes.

Manual counting of small species is presented as a task that requires the intervention of specialized personnel in addition to requiring a large amount of time. The effectiveness of the study, in terms of accuracy and reliability, is directly linked to the operator's experience and skills, as well as their level of fatigue. For example, when working with a sample plate with a diameter of 5 cm, it is necessary, either with a magnifying glass or a microscope, to zoom in on the sample in order to visualize the individuals present in it; Figure 1 displays mussel individuals in their early stage:

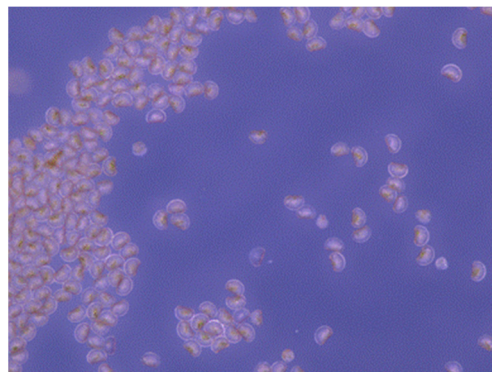


Figure 1. Mussel larvae at their 40-micrometer size, under microscope.

Counting all the individuals in Figure 1 may require three/five minutes from a qualified person; even more, the visible area in the figure represents a small percentage of the entire plate area (Figure 2). Consequently, the time to manually detect, classify, and count the individuals present in a small sample is incremented exponentially.

Sample:

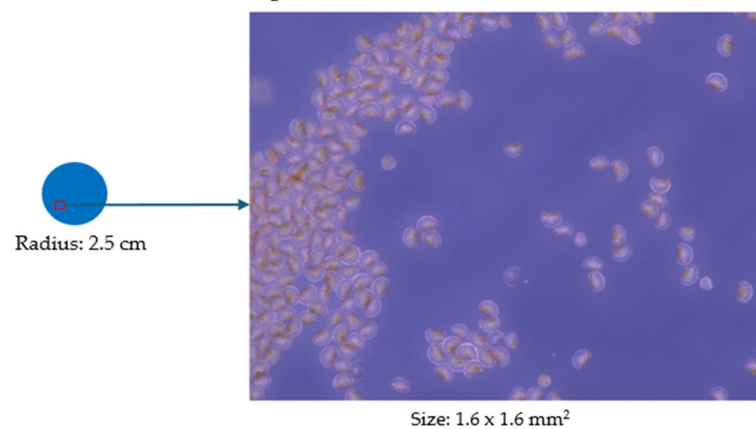


Figure 2. Laboratory test sample (a 2.5 cm radius circular plate). When observed through a microscope, 1.6×1.6 square millimeters is an area that allows manual inspection.

Computer vision is a part of artificial intelligence that aims to replicate the human capability to understand visual information. To achieve this, all that is needed is a computer

that can analyze the images and differentiate features in them that allow it to “understand” the objects present in them through algorithms that combine statistics, signal processing, and machine learning [7]. In this way, artificial vision can be considered an interesting tool to tackle tasks of this nature, to say the least. A robust machine vision model can drastically reduce the time it takes to perform these tasks while counteracting the fatigue experienced by operators. The high processing power of computer vision could be combined with the ability of specialized personnel to monitor predictions to speed up work and increase the number of hits.

Artificial neural networks are inspired by the neural structure of our brains. In a simplified way, the aim is for these neurons in the network to learn to solve the situations on which they have been inductively trained based on previous experience [8]. At each stage, when receiving mathematical information, these signals are weighted with different statistical methods. In this way, in the training stage of the network, the ‘weights’ of each neuron are adjusted so that the differences between the actual result and the result predicted by the network are minimized [9]. This is achieved by presenting pre-labeled data so that the network ‘learns’ from its errors through an iterative process in which the aforementioned weights for each neuron are adjusted in a graduated manner. Figure 3 shows the common structure of a neural network [10]:

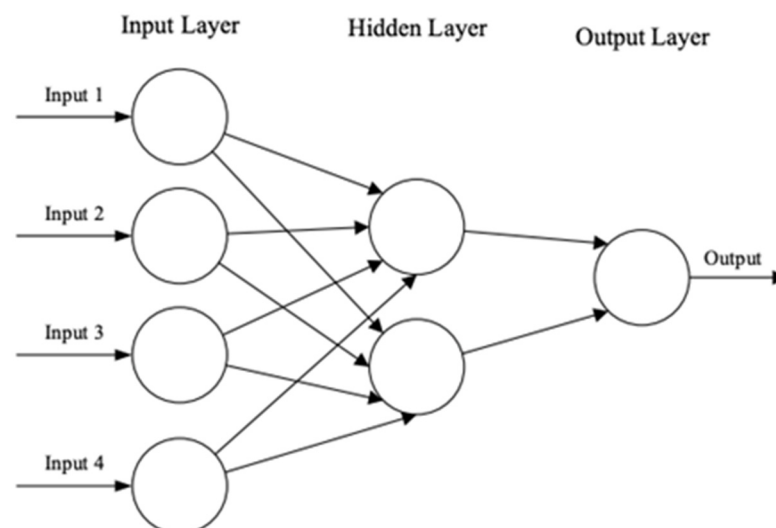


Figure 3. Structure of an artificial neural network: each neuron takes an input and applies an activation function to it. This input, depending on the layer we are in, will have different mathematical characteristics of the dataset or the output of a previous layer of the network. The neuron will apply an activation function to the sum of the weighted inputs and transmit the result to all the neurons in the next layer. To go from one neuron to another, a kind of ‘toll’ has to be paid, which are the weights mentioned above. Weights, therefore, control the strength of the connection between neurons, determining how much influence the input will have on the output. That is, when transmitting different inputs between neurons, the weights are applied together with an additional value called bias.

In the context of this work, the idea that the algorithm works with numerical parameters that guide the neural network towards the correct solution through predictions is taken up. The data passes through the network multiple times, which will cause a progressive refinement of the numerical values of the network until optimal results are reached, bringing us closer to the theoretically ideal parameters that cause a greater success rate. After the testing phase, the machine learning model is obtained, which will be able to perform the task for which it has been trained. The model is essentially composed of the numerical parameters obtained thanks to the algorithm and the Confidence for decision-making. The model represents the product obtained after the training of the algorithm and is the program used to perform the proposed task.

The idea of using computer vision and neural networks is not new and has been profusely used in the literature in other fields. In [11], different neural networks are used in series, combined with a feature vector that is made of image intensities sampler over a stencil neighborhood, proving advantages over previous membrane detection methods in the context of an automated system for the reconstruction of the connectome [11]. The authors of [12] use AI-aided cystoscopy to outperform urologists at recognizing and classifying bladder lesions using convolutional neural networks over the PubMed-MEDLINE database. The authors of [13], on the other hand, use supervised classification to compare histograms of oriented gradients and convolutional feature extractors in the field of detecting salmon muscle gaping using SVM with great results. A similar approach, but in a different context, can be found in [14], where authors address a key issue in neuroanatomy (segmentation of neuronal structures shown in electron microscopy images; the strategy here is to provide a pixel classifier to label each pixel whether it is a neuron or not, using convolutional and max-pooling layers that extract features, providing a probability of each class [14]. The authors of [15] automate the diagnostic process with the use of an intelligent system that would recognize malaria parasites, and aid efforts for the total elimination of malaria using microscopy and a specifically tailored CNN that outperforms VGGNet [15].

The proposed algorithm is based on the YOLO (You Only Look Once) method [16], a two-stage detector for objects in images. The initial phase of the method involves producing a broad array of region proposals for each image using either a heuristic algorithm or CNN [17], followed by the classification and regression of these identified candidate regions. In the context of object detection using computer vision, the main approaches followed in the literature are one and two-stage detectors [18].

YOLO is the most known example of a one-stage detector, which obtains a prediction of the existence of an object in a bounding box and classifies it into one of the different classes in one only inspection of the original image. Other well-known one-stage strategies are RetinaNet [19] and the Single Shot Multibox Detector (SSD) [20]. All one-stage strategies have in common that they divide the input image into cells to obtain a prediction of the subarea in the grid where the potential objects are present in their different classes, just using each cell at a time. They are particularly effective in real-time applications and commonly selected in low-computational scenarios because they are focused on the time needed to detect the objects.

As for two-stage detectors, although typically slower, they usually obtain more accurate object detections [21]. The main difference with one-stage mechanisms is how two-stage mechanisms propose the regions and how they classify the objects in their classes. In this case, the process is divided into two steps: first, the detector selects certain areas of the input image that have more likelihood of having the presence of the pursued object, using methodologies like RPN (region proposal networks) or selective search. Once the potential areas are identified, they are refined and classified to conclude whether there is an existence of the object and where. One-stage methods use the whole input image to obtain a prediction of the bounding boxes; two-stage mechanisms, on the other hand, perform a refinement of the selected areas of the input image (proposed in the first step) and accomplish the obtention of the bounding boxes and the classification based on the features found in the first stage. Two-stage methods do not focus on speed but accuracy is a common choice when computation resources are not an issue and the objects in the image present difficulties like overlapping or a high density [22]. In the context of this research, performing detection in a single step provided faster detection and allowed a simpler yet effective approach; the speed advantage was key to enabling real-time detection not only in still images but also in videos and life detections on the microscope.

Most mollusk identification processes rely on human visual ability and are an extensive and laborious task. This presents major limitations in terms of efficiency and scalability. In [23], the different techniques used in the scientific literature are shown. The morphological mechanisms using microscopic examinations are the primitive yet still more used method, despite the subjective nature of its philosophy (comparing images with voucher

collections and preserved specimens [23]. Due to being subjective, it is a method clearly prone to human error [23]. The immunological techniques, on the other hand, rely on unique ‘signature’ molecules that, once injected into a vertebrate host, will trigger an immune response that generates identifiable antibodies [23]; in this vein, polyclonal antibodies excel because of the higher affinity and wider reactivity, but it is an expensive method that only provides a markup for the individuals in the sample (keeping the classification according to size/age and account to the human operator). Finally, DNA-based systems use a combination of cytogenetics and cytology after dissolving the shell; this type of method is really accurate and bases its strategy on comparing a DNA sequence that is sampled from well-known identified cases (especially grown-up mussels), with the one extracted from the larvae to be identified. Again, this is a mechanism that not only takes time and money but also does not provide size classification according to age or automatic accounting [23].

Focusing on bivalve larvae, most of the literature is based on techniques based on image analysis (manual or automatic) [24]. The fact that both immunological and DNA-based mechanisms need specialized equipment not available in many laboratories, those methods based on images obtained through the microscope are coping with every case [24]. This reality brings the need for automatization of the inspection, detection, classification, and accounting of the species in the sample, and this is why different research for the development of intelligent systems is born [24]. To the best of our investigations, the automatized methods for mollusk larvae in the literature are scarce.

In the Mussel Classifier System Based on Morphological Characteristics [25], an automatic classifier of five species of mollusks based on their morphological characteristics is developed. This mimics the same procedure that skilled operators carry out to visually identify the different kinds of mollusks. The proposed system goes further by including statistical information on crop developments. With this system, a recognition rate of around 95% has been achieved, obtaining the results in a matter of seconds as long as the visual interface that demonstrates the results is not active.

Other studies focus on proposing solutions to prevent the settlement of invasive aquatic species. The following paper describes an autoencoder-based automated system for extracting features from image sets. An autoencoder is a type of neural network used to reduce and extract relevant features about a dataset. In other words, this technique is not designed to identify and classify different types of mollusks but to detect the differences between them in order to discriminate against invasive species. With this technique, a harmonic average between sensitivity and reliability of 97% has been achieved [26].

Another novel technique is the ShellBi method, which consists of the supervised classification of images by means of birefringence stops in the shells of bivalve larvae. The research seeks to improve the accuracy of ShellBi. The improvements have been obtained in larvae reared in conditions similar to those of the natural environment, so it concludes that it has application with field samples with a potential future to the extension of different marine systems [27].

In [28], an image processing algorithm is introduced to identify, classify, and count the mussels in seawater using computer vision, but the research is focused on grown mussels, not microscopic larvae. In [29], a CNN is proposed to identify and classify mussels from other types of species (barnacles, worms), but, again, it uses grown species instead of larvae under the microscope. The same approach is given in [30], where a CNN method is applied to perform image segmentation, with a genetic programming mechanism (but applied to buoys). Finally, in [31], Sentinel-2 images are used to automatically identify and count mussel platforms using MLP and a neural network.

Thus, different perspectives on the use of artificial vision and machine learning to address the identification of mollusks are highlighted but lack focus on larvae, especially mussel larvae. This proves that there is already evidence in the field and opens the way for us to carry out research in this field under a new premise. In addition, we will add a new approach by proposing an automatic counting of samples to speed up the work process of experts. The capabilities of this technology will also be explored to try to improve the

efficiency and accuracy in the identification of different classes according to the age of the same species of mollusk.

This research arises under the premise of implementing an artificial vision system for the detection, counting, and classification of bivalve larvae. The focus is on developing a computer vision model for the classification of cheek larvae according to their size; specifically, we classified them into 40, 100, and 180 μm (which are the average sizes of the individuals in their 3 key steps of mussel growth, and the data manufacturers need to understand when it is the right time to perform the insemination). The goal is, therefore, to develop, train, and test a specific computer vision model to differentiate the key characteristics of the mussel larvae individuals, so that automatic detection, accounting, and classification can be performed with no manual intervention in a small fraction of the time a human could perform this task. There are several premises that must be taken into account:

- The individuals to be detected can be in multiple positions and orientations, so a dataset will be needed, i.e., a set of images, in this case, large and varied enough to avoid overfitting the model.
- This technology has undergone multiple advances in recent years, but there is great difficulty when it comes to detecting small objects. To the untrained eye, the differences between the classes to be detected are not obvious. Being aware of this, we must establish some “limits” of our work and that is that we must always work with the same zoom and always under the most similar conditions possible to those that have been used to train the model.

As a result, the goals for carrying out the work were as follows:

- Implementation of an artificial vision system capable of analyzing images containing the different cheek larvae and distinguishing the characteristics for their classification.
- Creation of image data augmentation techniques to increase the number of samples to train the model.
- Evaluation of the model’s ability to perform, count, and classify the model, both by software and by traditional manual analysis techniques.
- Optimization of the model to maximize accuracy and efficiency in the identification of individuals, exploring possible adjustments and improvements in the algorithm.
- Develop a software tool for counting the predicted classes in order to be able to handle and work with the collected data.

These goals for the identification, accounting, and classification of mussel larvae are integrated within the purpose of achieving an artificial vision model capable of differentiating between an indeterminate variety of species and the species under study.

The manuscript is organized as follows: Section 2 describes the materials and methods that have been used both in the Marine Research Center belonging to Xunta de Galicia and the University of Vigo; Section 3 depicts the results obtained and provides a discussion; and finally, Section 4 introduces the conclusions.

2. Materials and Methods

An image is made up of pixels and in order to form these images, the spatial distribution of these pixels is important. That is why artificial neural networks are not an optimal architecture for developing computer vision models. This is how convolutional neural networks (CNNs) emerged, which are especially useful for images. The main characteristic of these networks is that they have many layers of convolutions. This makes it possible to differentiate features in a hierarchical way where each layer is good at detecting different shapes. The deeper the deeper the shapes, the more complex the shapes can be differentiated. CNN architectures are made up of 3 main layers:

- The convolutional layer: a convolution is a mathematical operation that combines two functions, giving rise to a third by superimposing and mixing the originals according to the equation:

$$(f * g)(t) = \int_{-\infty}^{\infty} f(\eta)g(t - \eta)d\eta \quad (1)$$

Input images are applied kernels or filters using the convolution operation, which leads to the highlighting of crucial features of the objects present in the image. This is because these operations generate convolutional feature maps that serve to highlight patterns (edges, shapes, or textures). After each convolution operation, a nonlinear activation function is applied to introduce nonlinearity factors and learn complex patterns.

The non-linear activation function that is applied is ReLU, which in a simplified way involves looking for the maximum value between an interval $(0, z)$. This function is very advantageous in order to avoid the disappearance of the gradient that is a relatively common problem during training phases, especially in those that have many layers. The problem is that the gradients of the weights become extremely small. This means that the adjustments of the values of the weights are negligible, that is, the values are barely being updated during training, resulting in a lack of learning of important characteristics. Another advantage of ReLU is that it is computationally less expensive than other activation features. The main disadvantages are that gradient fragility can occur when activations fall in the region where $x < 0$ occurs, causing the gradient to be null. This causes those particular neurons to stop responding to variations in input. Also, the activation range of ReLU is $[0, \infty)$ which means that there is a chance that the activation value will be triggered. Figure 4 displays the behavior of the ReLU function:

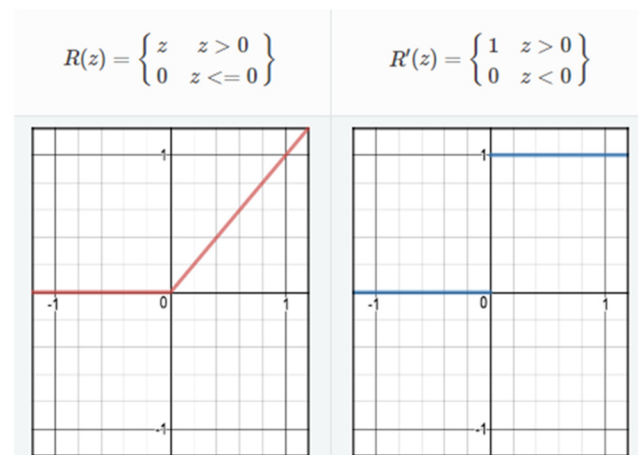


Figure 4. ReLU activation function.

- Pooling layer: the next layer performs a pooling operation, which reduces the size of feature maps and extracts features that are maintained over small shifts in the image. This operation reduces the number of parameters and computational complexity while preserving the essential characteristics to be differentiated.
- Fully connected layer: with the previous two layers, relevant features are obtained and fully connected layers are then used. These are responsible for classification and have connections between all the nodes in the previous layers. A softmax function is incorporated into them to generate ranking probabilities:

$$(f * g)(t) = \int_{-\infty}^{\infty} f(\eta)g(t - \eta)d\eta \quad (2)$$

The above outputs are transformed into normalized probabilities that add up to one, and the model's Confidence in the membership of each class is evaluated. In the function above, y represents the input vector and j represents the index of each class.

In order to accomplish real-time mussel larvae detection, accounting, and classification in their three primary categories, an evolution of the YOLO method has been developed. The strategy followed consists of adding a first step of attention methodology to the inner

network, which allows the network to find resources for the key features; then, the standard Mean Squared Error is used for the loss, resulting in great stability for the bounding box regression. The final step was to generate extra images from the original dataset using common augmentation techniques.

The process begun with the bounding boxes defined by 9 anchors that limit the area and the aspect ratio of the object. In order to establish the bounding box priors, clustering of k-means was used on the dataset, using the Euclidean distance metric as the algorithm for clustering. The idea behind k-means is that the data come from the k precise center plus certain noise in the Gaussian form. The algorithm initially selects k points (randomly) for centroid candidates, and then assigns every data point to their nearest cluster centroids. After that, the centroids are recalculated on the formed clusters after each iteration. This process of assigning and evaluating the centroids is repeated until the centroids are finally stabilized, or when a limit of iterations is reached. At last, the ultimate anchor size corresponds to the center of the polygon formed by the 9 clusters. This methodology helps alleviating the learning challenge for the model and improves the model's ability to detect bounding boxes.

For the context of identifying mussel larvae in their three primary stages (40, 100, and 180 μm), the images have to be obtained at a specific zoom factor, according to the zoom the model was trained; consequently, no change of scale was needed. The input image is divided into grids of $P \times P$, where the objects are to be detected and their bounding boxes need to be centered. Once divided in grids, the neural network generates the maps, extracting the features from each division. After that, the Feature Pyramid Network [32] employs upsampling of the output feature maps and subsequently concatenates them with the output from the preceding convolutional layer. The next step consists of a layer of attention, to focus the network on the key features for every channel. If we call F_{in} the $a \times p$ dimensions of the convolution layer with n channels, then the feature f_l of channel number l , having l in $\{1, \dots, n\}$, will be $F_{in} = \{f_1, f_2, \dots, f_l\}$. In every convolution, a squeeze operation is performed on F_{in} through the $a \times p$ dimensions, generating global features $r \in R^n$, encoding the features on each channel. Consequently, dimension feature number i can be calculated as follows:

$$r_i = F_e(f_i) = \frac{1}{a \times p} \sum_{i=1}^a \sum_{j=9}^p f_l(i, j) \quad (3)$$

From there on, the excitation was conducted on the global features. In order to capture the non-linear relation among the individual channels, we employed a straightforward gating method with an activation sigmoid function.

$$act = F_e(z, p) = \sigma(p_2(\tau p_1 r)) \quad (4)$$

σ is the Rectifier Linear Unit; p_1 represents the weight of the first fully connected layer that performs a compression in the parameters of the channels; p_2 denotes the weight of the second fully connected layer, to restore back the number of channels; and finally, the sigmoid function is represented by τ . At last, we propose the following:

$$\tilde{f}_l = F_{es}(f_l, y_l) = f_l \cdot act_d \quad (5)$$

Which is the scaling function, where the y_l parameter is a scalar that multiplies the feature of the d channel and obtains the output of this block of layers as $\{\tilde{f}_1, \tilde{f}_2, \dots, \tilde{f}_n\}$ for every channel.

As in the YOLO function, the function that evaluates the total loss can be calculated as the sum of the losses of each step (classification loss, Confidence loss, and bounding box

loss). The loss produced in the classification step can be obtained using categorical cross entropy, as given by the following:

$$L_{cl} = - \sum_{x=0}^{P^2} B_{xy}^m \sum_{g \text{ in classes}} [\widehat{Prob}_{xy} \text{Ln}(Prob_{xy}) + (1 - \widehat{Prob}_{xy}) \text{Ln}(1 - Prob_{xy})] \quad (6)$$

where B_{xy}^m denotes the y box preceding in the x grid cell ($P \times P$ grid); in case that preceding bounding box can predict the location of the target bounding box, B_{xy}^m will be 1, and zero in other case. \widehat{Prob}_{xy} represents the estimation of the probability that the larva belongs to class g, and $Prob_{xy}$ is the label of the ground truth.

The loss originated by the Confidence function has been selected as a modification of focal loss [33], which addresses the class imbalance by reshaping the standard cross entropy loss such that it down-weights the loss assigned to well-classified examples [33]; their strategy focuses training on a sparse set of hard examples and prevents the vast number of easy negatives from overwhelming the detector during training [34]. In our case, no weight was used to address the scale issue since the dataset and the samples were obtained using a specific zoom to handle the different sizes of the mussels in their growth; consequently, the proposed equation is as follows:

$$L_{co} = - \sum_{x=0}^{P^2} \sum_{y=0}^C [\beta(1 - C_{xy})^\delta \widehat{C}_{xy} \text{Ln}(\widehat{C}_{xy}) + (1 - \beta) C_{xy}^\delta (1 - \widehat{C}_{xy})] \quad (7)$$

where \widehat{C}_{xy} stands for the prediction of the Confidence, and C_{xy} is the Confidence of the ground truth box. The values of β and δ were obtained during the tests, with values of 0.75 and 2, respectively.

Finally, the loss of the localization of the bounding box prediction is given by MSE (Mean Squared Method), which quantifies the difference between the coordinates of the predicted bounding box and the coordinates of the ground truth bounding box. The operation is performed separately for the coordinates of the center of each bounding box and for the size of each bounding (for every bounding box found in the image). Using MSE, the average squared difference between the parameters of the bounding boxes (predicted and ground truth) can be estimated, using not only the accuracy of the location but also the accuracy of the size; minimizing this MSE value during the training step, the prediction mussel locations of the model is improved [35].

The dataset was also augmented in order to mitigate overfitting, by using two different mechanisms; the idea is to enhance the model's generalization capacity by using a greater quantity of images (since manual data collection and annotation involves an important amount of effort, we expanded the dataset).

The first data augmentation technique used was CutMix [36]. This technique, instead of just deleting pixels, substitutes the erased areas with small patches sourced from other images. At the same time, the ground truth labels are adjusted in proportion to the cumulative pixel count of the merged images. This methodology allows us to avoid pixels with little to no information during training, thereby improving training efficiency, while still keeping the benefits of regional dropout. Moreover, the introduced patches augment the model's localization ability by necessitating the identification of objects from partial perspectives [36]. In this research, CutMix provided different contexts that improved our generalization capabilities, minimized overfitting, and worked as a regularizer; it provided identification even when the larvae were under other ones, and reduced the time to create the dataset database.

The second data augmentation technique used was a rotation of the image. The rationale is the same: the number of images available for training is limited, so it is interesting to use data augmentation techniques to increase the number of images we have. The goal is to improve the generalizability and performance of the models we are going to develop. The technique consists of applying different manipulations and transformations

to existing images to generate new data based on previous ones. Given the conditions of the research, we have decided to rotate the images we have. This makes sense if we think that the individuals present in a sample can be found in any position: turned, one on top of the other, etc.

Although this task may seem trivial, we must bear in mind that at the same time as the images are rotated, we must rotate the coordinates of all the bounding boxes and store them in new text files to save the new information. We generated the corresponding rotation matrices and forced the same scale to be maintained during the rotation. By doing so, certain areas in the images will remain empty; to solve this issue, mirror techniques have been created and used so as not to leave empty areas. These reflections must also be taken into account when creating the new files with the coordinates of the bounding boxes, as there may be detectable individuals that we must tell the algorithm that they are there. The rotation was performed using OpenCV, following the next steps:

1. Obtention of the labels in the image (transforming the coordinates to the OpenCV format).
2. Rotation of the images: first, a rotation matrix is obtained for every angle in the 0–360° interval, with a 2° step size; then, the image is rotated.
3. Bounding-boxes translations: whenever the angle is different from 0°, the points are moved using the rotation matrix; if the corresponding bounding box is not moved away from the limits of the image, the change is made permanent, and the coordinates are translated to the YOLO format again.

Using CutMix and rotation, the dataset obtained was balanced, contained enough images to perform the training and validation steps, and provided enough context to achieve good results.

3. Results and Discussion

To perform the training and the testing of the model, two datasets were created:

- The first one was used for training and validation purposes. To create it, 6400 images of 2.5 cm radius circular laboratory plates were taken, each containing seawater captured in a mussel sea farm. In the laboratory of the fishing farm, mussels are kept in separate seawater tanks according to their size and gathered using sleeve filters for each specific size of each class (40, 100, 180); with this procedure, it can be guaranteed that each sample just contains individuals of one class, what helps labeling the images. Table 1 displays the samples obtained for the first dataset:

Table 1. Dataset description.

Class	Size (μm)	Samples
0	40	2400
1	100	1800
2	180	1200

Table 2 shows the characteristics of the optics used to take the images of the samples; each sample was photographed twice (in two different microscope configurations) to handle the different larvae sizes:

Table 2. Properties of the image capturing process in their two configurations.

Configuration	Zoom	Video Magnification	Objective	Aperture
1	4.99×	1.6×	1×/0.09	0.09302
2	3.97×	1.3×	1×/0.08	0.07673

- The second dataset was formed by mixing individuals of the three different classes in a tank of seawater so that it would contain mussels at three different stages in their growth. This dataset was used for validation purposes, to detect, classify, and count the individuals.

Once we had a relatively large dataset, it was labeled to inform the algorithm about the different classes present in the samples. For this process, an open-source third-party tool, *labellmg*, was used. The bounding boxes around each mussel were manually identified by a team of biology experts, resulting in a text file in which the class is identified with an id (in this case 0 = 40 μm , 1 = 100 μm and 2 = 180 μm) along with the 4 points that make up the vertices of the boxes within the images. Figure 5 showcases the process of manually labeling the samples in the pictures taken:

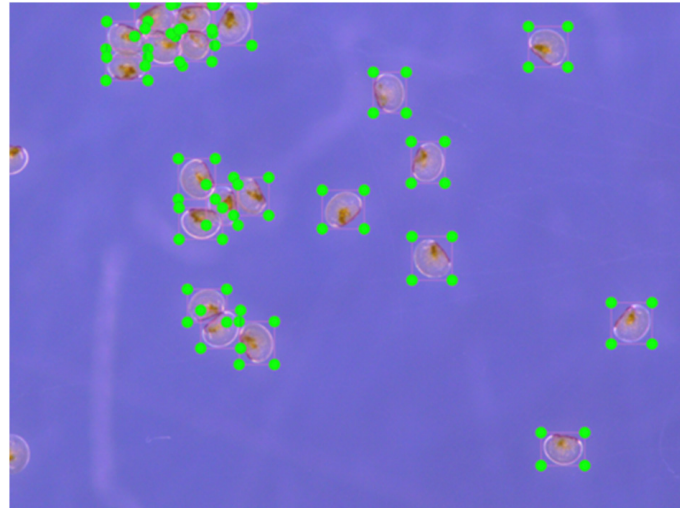


Figure 5. Manual labeling of the samples.

The next step was to perform the data augmentation described in the previous section and to save 20% of the images taken for the testing phase of our model and keep 80% of them for training purposes. A Python script was created to perform the CutMix algorithm and to distribute the files randomly in a *train* and *test folder* according to the use they would have. Special care was taken to make sure that a similar number of images of all classes were available in order to realistically evaluate the model under random circumstances.

Figure 6 displays a picture of a sample of seawater with mussel larvae to test the capability of the algorithm:

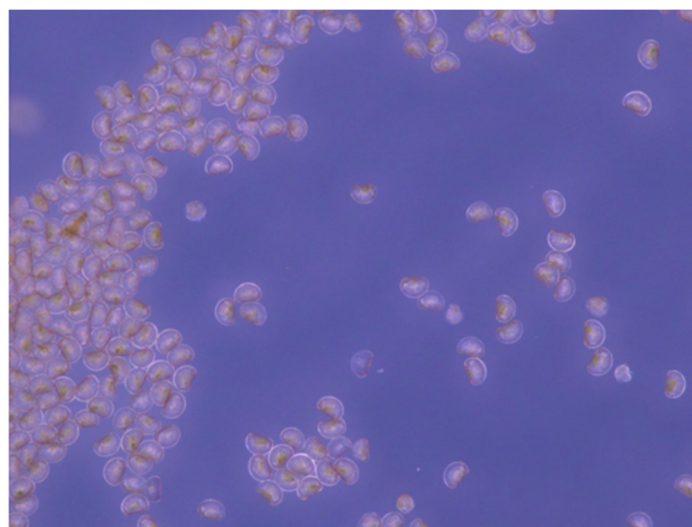


Figure 6. Counting mussel larvae in a sample plater under microscope.

The algorithm was implemented under the Python language. Table 3 displays the average number of individuals accounted in different pictures of different mussel classes,

and the average manually accounted larvae present, as well as the time spent in both types of processes. It is shown that the average time the algorithm takes to evaluate the presence of the larvae remains practically constant, whilst the average time a scientist needs to account for the individuals is much longer; it is necessary to remark, also, that these samples just contain larvae of the same class, making it much easier for a human to make the accounting.

Table 3. Average manual vs automatic accounting.

Class	Size (μm)	Manually		Automatic	
		#	Time (s)	#	Time (s)
0	40	204	383	206	9
1	100	84	141	83	8
2	180	49	77	51	8

Another test performed to check the validity of the algorithm was to use pictures of other species, very similar in their larva state, to mussel: clam. Figure 7 displays a picture under the same zoom conditions and for individuals of the same size ($40\ \mu\text{m}$).

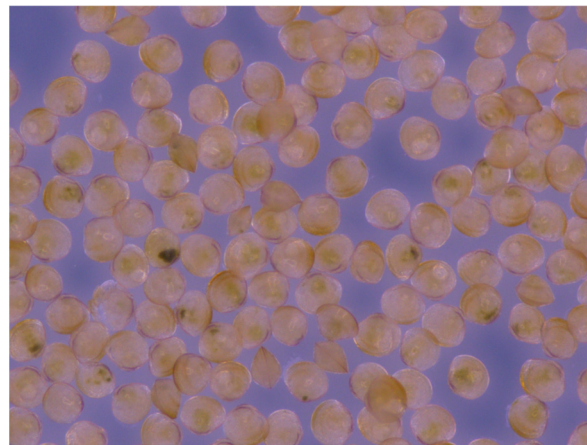


Figure 7. Clam larvae of $40\ \mu\text{m}$ under the same zoom conditions.

Table 4 shows the results of the test where the clam was introduced to check whether the algorithm identified it as a mussel or not. As the table displays, the algorithm correctly identified no mussel larvae in the same; indeed, although the larvae are very similar to the human eye, in this sample there were no mussel larvae (consequently, the algorithm proved to be working properly under this test).

Table 4. Clam larvae of $40\ \mu\text{m}$ to test if they were identified wrongly as a mussel.

Class	Size (μm)	Manually		Automatic	
		#	Time (s)	#	Time (s)
0	40	183	305	0	6

Additionally, a set of tests were performed using several samples containing a mix of different larvae (in different stages of their growth) in different sizes. Figure 8 displays a picture of a sample used in the laboratory where the algorithm has found the individuals, classified the different classes, and accounted for the number of each class.

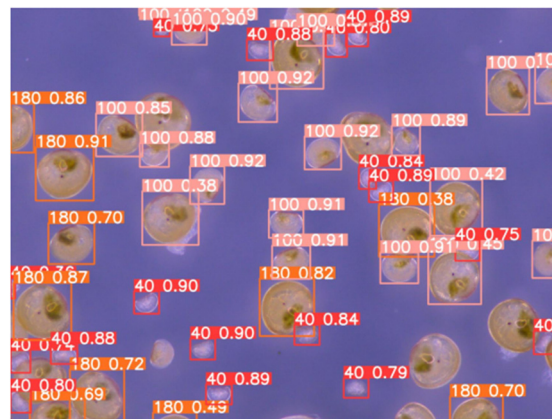


Figure 8. Seawater sample with mussels in different stages (sizes).

Finally, a final set of tests were carried out to determine the effectiveness of the algorithm, in terms of not confusing another species incorrectly as if they were mussels. In the laboratory, a dataset of samples that had fine clams were prepared, which are very similar to mussels in their larva state. Figure 9 displays the results of the detection of fine clams at their 40–100–180 μm stage of growth: as no bounding boxes are shown, the algorithm confirms that no individuals of mussel larvae are present in the sample (in which it is really difficult for the eye to distinguish between them).

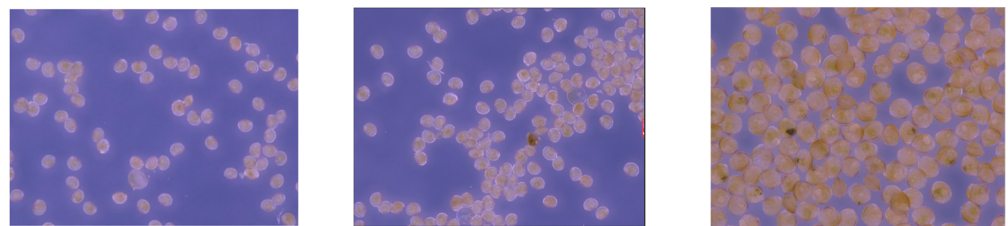


Figure 9. Fine clam under the microscope, in their 40 (left), 100 (center), and 180 (right) μm stages of growth. The algorithm correctly identified no mussel larvae, although the similarities in size and shape are clear.

The metrics used to evaluate the algorithm include Precision and the rate of Recall (P and R , respectively). They both are based on true positives (TP , meaning a correct detection of a ground-truth bounding box), false positives (FP , a false detection of a non-existing object within the image), and false negatives (FN , an undetected ground-truth bounding box) [37]. In this case, we opted to define as positive samples those with an overlapping between the predicted bounding box and the ground truth (IoU) of over 0.5.

Precision in computer vision is the ability that a model has to identify just relevant objects in an image (in percentual terms, the percentage of correct positive predictions). Recall refers to the capability of the algorithm to find all relevant cases (every ground truth bounding box in the image); it can be written as the percentage of correct positive predictions among every give ground truth:

$$P = \frac{TP}{TP + FP} \quad (8)$$

$$R = \frac{TP}{TP + FN} \quad (9)$$

Moreover, the average position was employed to determine the algorithm's performance in terms of detection, calculated as follows:

$$AP = \int_{R=0}^1 P(R).dR \quad (10)$$

Finally, mean average recision explains the average precision (AP) calculated not for every class (size of the mollusk), but for every class under study:

$$mAP = \frac{1}{g} \sum_i^g AP_i \quad (11)$$

Figure 10 provides a detailed visualization of the Recall–Confidence Curve for the various classes, serving as a critical indicator of the algorithm's performance. The curve illustrates how the algorithm's Recall changes as the Confidence threshold is adjusted. Recall, defined as the ratio of true positive instances to the sum of true positives and false negatives, measures the ability of the algorithm to identify all relevant instances. The curve's shape and the specific data points offer insights into the balance between correctly identified instances and the Confidence level at which these identifications occur. This shows the algorithm's efficacy in recognizing different classes under varying degrees of Confidence.

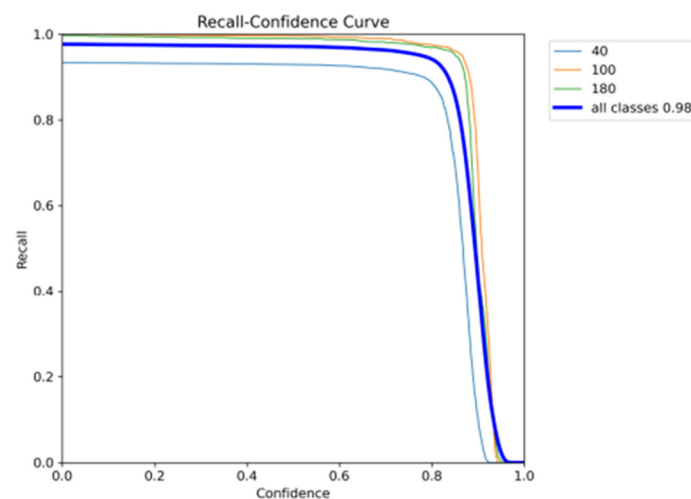


Figure 10. Recall–Confidence Curve.

The results highlighted in Figure 10 demonstrate that the algorithm achieves a high level of accuracy. When applying a reasonable Confidence threshold, it can be observed that more than 90% of the classes were correctly identified. This high Recall rate indicates that the algorithm is performing as expected in detecting the majority of relevant instances within each class, thereby ensuring robust performance across different categories. The implication is that the algorithm maintains a strong ability to discern relevant data points even when adjusting for confidence levels, reflecting its overall reliability and precision in practical applications.

To further enhance the reliability of the algorithm and mitigate the risk of false positives, predictions with confidence levels below 50% were systematically discarded. This threshold was established to ensure that only those predictions with a higher likelihood of accuracy were considered. By setting this cutoff, the number of incorrect classifications, or false positives, was significantly reduced. This filtering mechanism ensures that the algorithm's outputs are not only accurate but also trustworthy, thereby increasing the overall integrity of the classification results. The balance between Recall and Confidence underscores the algorithm's capability to deliver high-quality predictions while minimizing errors, making it a valuable tool for classifying the individuals in a water sample.

Figure 11 illustrates the Precision–Recall curve, a tool for evaluating the performance of the model in terms of its ability to correctly identify positive instances. Precision, defined as the number of true positives divided by the sum of true positives and false positives, measures the proportion of positive identifications that were accurately detected by the algorithm. This metric helps us to understand how well the model is distinguishing between true positive cases and false alarms. The Precision–Recall Curve in Figure 11 provides a visual representation of this relationship across different thresholds, highlighting the model’s efficacy in maintaining high precision while adjusting recall levels.

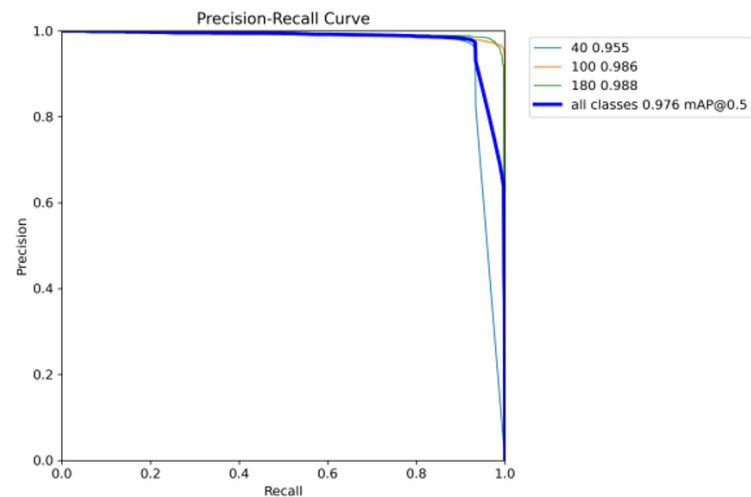


Figure 11. Precision–Recall Curve.

The curve indicates that the model demonstrates an average Precision value of 97.6%, showing its capability to correctly identify positive instances with minimal errors. This high Precision rate suggests that the model is avoiding incorrect labeling of negative cases, thus reducing the occurrence of false positives. The ability to achieve this Precision is indicative of the model’s robustness, particularly in scenarios where accurate detection of positive cases is critical. Overall, the Precision–Recall Curve not only reflects the model’s strong performance but also its capacity to deliver precise classifications, making it a valuable asset in the laboratory.

The F1 score was calculated, providing an evaluation metric that harmonizes the average between Precision and Recall. The F1 score is particularly interesting in scenarios where there is an imbalance in the sample sizes among different classes, as it accounts for both the Precision (the ratio of true positive instances to the sum of true positives and false positives) and the Recall (the ratio of true positive instances to the sum of true positives and false negatives). By combining these two metrics, the F1 score offers a balanced measure that reflects the model’s ability to identify relevant instances while also minimizing the occurrence of false positives and negatives.

Figure 12 presents the F1–Confidence Curve, which is an indicator of the optimal Confidence threshold to use for minimizing errors. This curve illustrates how the F1 score varies with different Confidence levels, guiding the selection of a threshold that balances Precision and Recall effectively. By identifying the appropriate Confidence level, the model can reduce both false positive and negative occurrences, enhancing overall prediction accuracy. This approach ensures that the model performs robustly, even when trained on an unbalanced dataset, thereby providing reliable and precise classifications across diverse classes.

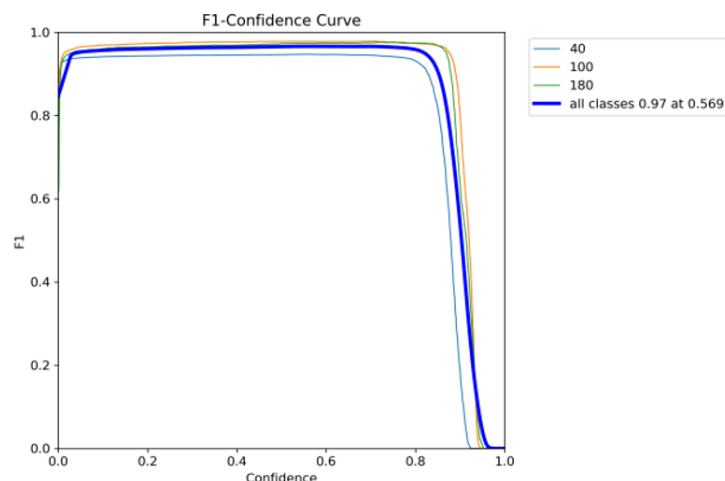


Figure 12. F1-Confidence Curve.

4. Conclusions

The European Union's mussel production industry is a significant market, dependent on obtaining mussel larvae as seed for cultivation. Traditionally, the process of monitoring larvae presence in seawater involves manual sampling and labor-intensive microscopic counting, susceptible to human error and time-consuming procedures. To address these challenges, our research introduces a computer vision-based methodology aimed at accurately identifying, classifying, and quantifying mussel larvae individuals across various developmental stages from microscopic images of water samples.

Our approach is the utilization of a neural network architecture derived from the YOLO method, incorporating convolutional, pooling, and fully connected layers to automate the detection, classification, and accounting tasks. Through the use of several manually labeled samples for model training and evaluation, along with data augmentation techniques to mitigate overfitting risks, we established a robust framework capable of effectively processing diverse larval specimens.

The outcomes of our study highlight the effectiveness of the proposed algorithm, as evidenced by standard performance metrics including Recall–Confidence, Precision–Recall, and F1 scores. These metrics affirm the algorithm's aptitude in accurately identifying, classifying, and accounting for different mussel larval classes present in microscopic images, surpassing human capacities in both speed and reliability. Consequently, our research not only presents a viable solution to streamline mussel larvae monitoring processes but also showcases the potential of computer vision techniques in enhancing efficiency and accuracy within aquaculture industries.

The next steps in our research include extending the spectrum of species to the rest of the bivalve mollusks under industrial growth (not only mussels but also clams, oysters, cockles, razor clams, etc.), and developing an automated machine that will accept laboratory plates with seawater and will perform the capture of the image and return the accounting of the different species individuals.

Author Contributions: Conceptualization, P.O.-C. and C.G.-S.; data curation, G.R.-G.; formal analysis, F.A.-A.; funding acquisition, C.G.-S.; investigation, P.O.-C.; methodology, P.O.-C.; project administration, F.A.-A.; resources, C.G.-S., F.A.-A. and G.R.-G.; software, P.O.-C.; supervision, F.A.-A.; validation, C.G.-S. and G.R.-G.; writing—original draft, P.O.-C.; writing—review and editing, G.R.-G. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the General Directorate of Fisheries, Aquaculture and Technological Innovation (Consellería do Mar) and by the European Union—FEMP. Measure 2: Increase in the potential of aquaculture production areas (Art. 51). Co-financing percentage measured 75% EMFF and 25% Member State (EMFF: European Maritime and Fisheries Fund).

Data Availability Statement: The data presented in this study are available on request from the corresponding author due to privacy reasons.

Acknowledgments: Ana Cerviño Otero, José Alberto de Santiago Meide, María Victoria Gregorio Chenlo and Susana Novoa Vázquez, for the selection of the samples.

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

References

1. Aquaculture Statistics. (n.d.). Ec.europa.eu. Available online: https://ec.europa.eu/eurostat/statistics-explained/index.php?title=Aquaculture_statistics#EU_Aquaculture (accessed on 4 April 2024).
2. Labarta, U.; Fernández-Reiriz, M.J.; Pérez-Camacho, A.; Pérez-Corbacho, E. (Eds.) *Bateiros, Mar, Mejillón. Una Perspectiva Bioeconómica*; Editorial Galaxia: Vigo, Spain, 2004; pp. 19–47.
3. Fuentes, J.; Molares, J. Settlement of the mussel *Mytilus galloprovincialis* on collectors suspended from rafts in the Ría de Arousa (NW of Spain): Annual pattern and spatial variability. *Aquaculture* **1994**, *122*, 55–62. [[CrossRef](#)]
4. Pérez-Camacho, A.; Labarta, U.; Beiras, R. Growth of mussels (*Mytilus edulis galloprovincialis*) on cultivation rafts: Influence of seed source, cultivation site and phytoplankton availability. *Aquaculture* **1995**, *138*, 349–362. [[CrossRef](#)]
5. Babarro, J.M.F.; Reiriz, M.J.F.; Labarta, U. Growth of seed mussel (*Mytilus galloprovincialis* Lmk): Effects of environmental parameters and seed origin. *J. Shellfish Res.* **2000**, *19*, 187–193.
6. Babarro, J.M.F.; Labarta, U.; Fernández-Reiriz, M.J. Growth patterns in biomass and size structure of *Mytilus galloprovincialis* cultivated in the “Ría de Arousa” (north-west Spain). *J. Mar. Biol. Assoc.* **2003**, *83*, 151–158. [[CrossRef](#)]
7. Szeliski, R. *Computer Vision: Algorithms and Applications*; Springer Nature: Cham, Switzerland, 2022.
8. Abdolrasol, M.G.; Hussain, S.S.; Ustun, T.S.; Sarker, M.R.; Hannan, M.A.; Mohamed, R.; Ali, J.A.; Mekhilef, S.; Milad, A. Artificial neural networks based optimization techniques: A review. *Electronics* **2021**, *10*, 2689. [[CrossRef](#)]
9. Kujawa, S.; Niedbała, G. Artificial neural networks in agriculture. *Agriculture* **2021**, *11*, 497. [[CrossRef](#)]
10. Chen, Y.; Song, L.; Liu, Y.; Yang, L.; Li, D. A review of the artificial neural network models for water quality prediction. *Appl. Sci.* **2020**, *10*, 5776. [[CrossRef](#)]
11. Jurrus, E.; Paiva, A.R.; Watanabe, S.; Anderson, J.R.; Jones, B.W.; Whitaker, R.T.; Jorgensen, E.M.; Marc, R.E.; Tasdizen, T. Detection of neuron membranes in electron microscopy images using a serial neural network architecture. *Med. Image Anal.* **2010**, *14*, 770–783. [[CrossRef](#)] [[PubMed](#)]
12. Negassi, M.; Suarez-Ibarrola, R.; Hein, S.; Miernik, A.; Reiterer, A. Application of artificial neural networks for automated analysis of cystoscopic images: A review of the current status and future prospects. *World J. Urol.* **2020**, *38*, 2349–2358. [[CrossRef](#)] [[PubMed](#)]
13. Xu, J.L.; Sun, D.W. Computer vision detection of salmon muscle gaping using convolutional neural network features. *Food Anal. Methods* **2018**, *11*, 34–47. [[CrossRef](#)]
14. Ciresan, D.; Giusti, A.; Gambardella, L.; Schmidhuber, J. Deep neural networks segment neuronal membranes in electron microscopy images. In *Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Glasgow, UK, 2012; Volume 25.
15. Peñas KE, D.; Rivera, P.T.; Naval, P.C. Malaria parasite detection and species identification on thin blood smears using a convolutional neural network. In Proceedings of the 2017 IEEE/ACM International Conference on Connected Health: Applications, Systems and Engineering Technologies (CHASE), Philadelphia, PA, USA, 17–19 July 2017; pp. 1–6.
16. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
17. Gu, J.; Wang, Z.; Kuen, J.; Ma, L.; Shahroudy, A.; Shuai, B.; Liu, T.; Wang, X.; Wang, G.; Cai, J.; et al. Recent advances in convolutional neural networks. *Pattern Recognit.* **2018**, *77*, 354–377. [[CrossRef](#)]
18. Li, Z.; Liu, F.; Yang, W.; Peng, S.; Zhou, J. A survey of convolutional neural networks: Analysis, applications, and prospects. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, *33*, 6999–7019. [[CrossRef](#)] [[PubMed](#)]
19. Wang, Y.; Wang, C.; Zhang, H.; Dong, Y.; Wei, S. Automatic ship detection based on RetinaNet using multi-resolution Gaofen-3 imagery. *Remote Sens.* **2019**, *11*, 531. [[CrossRef](#)]
20. Magalhães, S.A.; Castro, L.; Moreira, G.; Dos Santos, F.N.; Cunha, M.; Dias, J.; Moreira, A.P. Evaluating the single-shot multibox detector and YOLO deep learning models for the detection of tomatoes in a greenhouse. *Sensors* **2021**, *21*, 3569. [[CrossRef](#)] [[PubMed](#)]
21. Carranza-García, M.; Torres-Mateo, J.; Lara-Benítez, P.; García-Gutiérrez, J. On the performance of one-stage and two-stage object detectors in autonomous vehicles using camera data. *Remote Sens.* **2020**, *13*, 89. [[CrossRef](#)]
22. Cazzato, D.; Cimarelli, C.; Sanchez-Lopez, J.L.; Voos, H.; Leo, M. A survey of computer vision methods for 2d object detection from unmanned aerial vehicles. *J. Imaging* **2020**, *6*, 78. [[CrossRef](#)] [[PubMed](#)]
23. Garland, E.D.; Zimmer, C.A. Techniques for the identification of bivalve larvae. *Mar. Ecol. Prog. Ser.* **2002**, *225*, 299–310. [[CrossRef](#)]
24. Hendriks, I.E.; Van Duren, L.A.; Herman, P.M. Image analysis techniques: A tool for the identification of bivalve larvae? *J. Sea Res.* **2005**, *54*, 151–162. [[CrossRef](#)]
25. Coelho-Caro, P.A.; Saavedra-Rubilar, C.E.; Saforilli, J.P.; Gallardo-Nelson, M.J.; Guaquin, V.; Tarifeño, E. Mussel Classifier System Based on Morphological Characteristics. *IEEE Access* **2018**, *6*, 76935–76941. [[CrossRef](#)]

26. Chowdhury, S.; Hamerly, G. Recognition of Aquatic Invasive Species Larvae Using Autoencoder-Based Feature Averaging. In *International Symposium on Visual Computing; Advances in Visual Computing*; Springer International Publishing: Cham, Switzerland, 2022; Volume 13598, pp. 145–161.
27. Goodwin, J.D.; North, E.W.; Thompson, C.M. Evaluating and improving a semi-automated image analysis technique for identifying bivalve larvae. *Limnol. Oceanogr. Methods* **2014**, *12*, 548–562. [[CrossRef](#)]
28. Zeng, D.; Liu, I.; Bi, Y.; Vennell, R.; Briscoe, D.; Xue, B.; Zhang, M. A new multi-object tracking pipeline based on computer vision techniques for mussel farms. *J. R. Soc. N. Z.* **2023**, 1–20. [[CrossRef](#)]
29. Signor, J.; Schoefs, F.; Quillien, N.; Damblans, G. Automatic classification of biofouling images from offshore renewable energy structures using deep learning. *Ocean. Eng.* **2023**, *288*, 115928. [[CrossRef](#)]
30. Bi, Y.; Xue, B.; Briscoe, D.; Vennell, R.; Zhang, M. A new artificial intelligent approach to buoy detection for mussel farming. *J. R. Soc. N. Z.* **2023**, *53*, 27–51. [[CrossRef](#)]
31. Martin-Rodriguez, F.; Isasi-de-Vicente, F.; Fernández-Barciela, M. Automatic Census of Mussel Platforms Using Sentinel 2 Images. *arXiv* **2022**, arXiv:2204.04112.
32. Pan, H.; Chen, G.; Jiang, J. Adaptively dense feature pyramid network for object detection. *IEEE Access* **2019**, *7*, 81132–81144. [[CrossRef](#)]
33. Lin, T.Y.; Goyal, P.; Girshick, R.; He, K.; Dollár, P. Focal loss for dense object detection. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2980–2988.
34. Li, X.; Wang, W.; Wu, L.; Chen, S.; Hu, X.; Li, J.; Yang, J. Generalized focal loss: Learning qualified and distributed bounding boxes for dense object detection. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 21002–21012.
35. Mao, Q.C.; Sun, H.M.; Liu, Y.B.; Jia, R.S. Mini-YOLOv3: Real-time object detector for embedded applications. *IEEE Access* **2019**, *7*, 133529–133538. [[CrossRef](#)]
36. Yun, S.; Han, D.; Oh, S.J.; Chun, S.; Choe, J.; Yoo, Y. Cutmix: Regularization strategy to train strong classifiers with localizable features. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 6023–6032.
37. Padilla, R.; Passos, W.L.; Dias, T.L.; Netto, S.L.; Da Silva, E.A. A comparative analysis of object detection metrics with a companion open-source toolkit. *Electronics* **2021**, *10*, 279. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.