Object tracking-based deep learning for calculating the quantity of eel seedlings (*Anguilla bicolor*)

Septi Aleyda, *Indra* Jaya*, and *Muhammad* Iqbal

Department of Marine Science and Technology, Faculty of Fisheries and Marine Science, IPB University, 16680, Bogor, West Java, Indonesia

Abstract. *Anguilla bicolor* is a highly economically valuable species of eel, whose cultivation needs to be carried out to the maximum. One of the factors supporting successful aquaculture is the optimal stocking density. The current calculation method for determining seedling quantity for stocking density is ineffective for large-scale cultivation. Therefore, in this study, an object-tracking-based calculation method using deep learning was applied to address this issue. Deep learning using YOLO v8 is an advanced method for counting and detecting objects including fish. This algorithm was used with the addition of the ByteTrack algorithm for object tracking and supervision for object counting. Object tracking is applied such that during seed counting, moving objects can be tracked and not counted repeatedly. Training was conducted on two datasets with different labels, namely the head and tail parts, using 300 epochs, 16 batches, and learning rate of 0,001 and 0,01. Based on the training results, the model with the head label has higher precision and recall than the tail-labelled model, with values of 0,91 and 0,93 respectively. The model performed well at a stocking density of 63 individuals/m², with accuracy and F1 scores of 90,91% and 0,95 respectively.

1 Introduction

Anguilla bicolor, also known as Indonesian shortfin eel, is a promising fishery commodity. The increasing demand for this eel, coupled with its high economic value, supports this prospect. This species is one of the eels that inhabit Indonesian waters and is not covered by the Convention on International Trade in Endangered Species of Wild Fauna and Flora (CITES) as a protected species. This has led eel-consuming countries to switch to this species [7]. The international demand for eel is 300,000 tons per year, with prices ranging from Rp180,000-225,000/kg, while in the national market (DKI Jakarta), it is around 3 tons per month, with prices reaching Rp150,000/kg [14]. However, meeting this demand is still difficult because eel seedlings rely on natural catches. Therefore, eel cultivation efforts continue to be made to meet this demand.

One of the factors that determines the success of eel cultivation is the availability of seedlings in terms of quality, quantity, and continuity [12]. Abundant seedlings with high

* Corresponding author : indrajaya@apps.ipb.ac.id

quality and continuous supply are the key to successful cultivation. This can be achieved through optimal stocking density for eel growth and survival [13]. The number of seedlings can be counted manually or by using automated technology, such as the light sensors used by [3] and [2] to count catfish and milkfish seedlings. Manual counting is currently not effective on a large scale, as it takes a lot of time, and the slimy body of the eel makes direct contact difficult for breeders. Visual manual counting without direct contact also faces difficulties in tracking, as eel seedlings constantly move in containers, leading to repetitive counting. Existing automated methods are limited to detection using light sensors. These sensors rely solely on colour as a detection parameter, and the data tends to be unstable at different distances, resulting in significant errors.

Deep learning is an advanced method for counting and detecting objects. Over the past few years, deep learning has significantly affected how the world is embracing Artificial Intelligence. One of the well-known object detection algorithms is You Only Look One (YOLO), which is preferred for its speed. Deep learning achieves efficient object detection and tracking without sacrificing the performance. It involves the use of neural networks and multiple layers to establish an appropriate formula [4]. Therefore, deep learning is an advanced method for counting objects, including fishes.

The application of deep learning involves several steps. Object detection is the initial step, in which algorithms such as YOLO v8 are used for accurate detection [4]. Object tracking is then applied to monitor moving objects. Once an object is detected, it is assigned a marker/ID and it enters the counting stage. Setting boundaries on an object's shape or movement can aid tracking [8]. In the case of eel seedlings, their heads and tails are chosen as distinctive features for detection because of their consistent shape. The authors of this study created a deep learning model that included the head and tail of the eel seedlings as a label for object detection. This study serves as a starting point for implementing deep learning-based object tracking to automatically count eel seedlings. The distinctive features of the head and tail impact the accuracy and performance of object-tracking calculations. The model developed using architectures from YOLO v8, ByteTrack, and Supervision can be applied in computer vision or other camera detection devices for faster and more accurate counting of eel seedlings.

2 Methods

2.1 Dataset collection

The seedlings used in this study were eel seedlings of the species *Anguilla bicolor* in the elver phase. This phase considered to be the most suitable because it has already unique visual characteristics that computer can detect as a ‘label’. The seedlings undergoing the grading process were recorded using video capture. These videos include variations in the environment and number of fish seedlings. The environmental variations included videos with white (**Fig. 1, Fig. 2, and Fig. 3**) and greenish backgrounds (**Fig. 4**). Eel seedlings in small containers or with a white background have variations in the number of seedlings, consisting of low, medium, and high densities as shown in **Fig. 1, Fig. 2, and Fig. 3**. The videos were recorded in containers of different dimensions and densities. The recorded videos were then extracted into 50 individual images that were used as the dataset for further analysis. After the preprocessing and augmentation process, the final dataset that will go through the training process amounts to 1000 images with a ratio of 800 images for training and 200 images for validation.

Fig 1. Low density.

Fig 2. Medium density.

Fig 3. High density.

Fig 4. Greenish background.

2.2 Data labelling

The preparation of the dataset was followed by its labelling. Two types of datasets were produced: datasets with labels on the head and datasets with labels on the tail. These labels are considered to be the visual variables in this research, with the objective of determining the best body part for object detection on the seedling. The head included the upper head of the pre-dorsal head (**Fig. 5**). Meanwhile, the tail part is taken in one of the places between the tip of the tail fin and the anodorsal part (**Fig. 6**) [6].

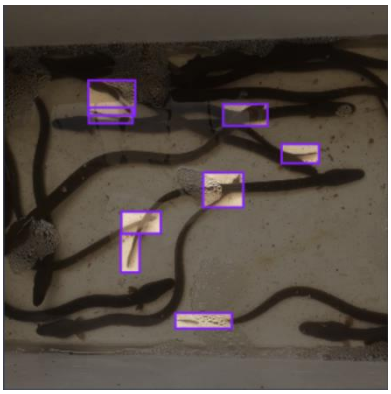


Fig 5. Label on the seedling's tail.

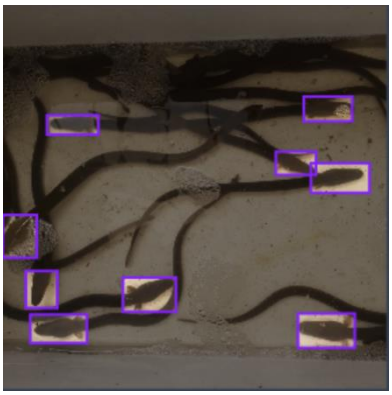


Fig 6. Label on the seedling's head.

2.3 Training, testing, and evaluation

The training stages were performed using a labelled dataset. The YOLO v8 deep learning algorithm was obtained from https://github.com/roboflow/notebooks/blob/main/notebooks/train-yolov8-object-detection-on-custom-dataset.ipynb. Training a deep learning model involves feeding data and adjusting its parameters to make accurate predictions. The parameters used were 300 epochs, 16 batches, patience 300, learning rate 0.001 and 0.01, and image size 800 × 800 pixels. Batch is the number of samples used in one iteration [11]. The learning rate influences the speed of neural networks in minimum solutions. Epoch is the number of algorithm iterations during the training process. Despite using 300 epochs, YOLO v8 has an early stop function that is useful for preventing overfitting by stopping the training process if the accuracy value is repeatedly the same [9]. The early stop function can be undone using the patience function.

After training, the training results were validated to determine the ability of the trained model to recognize an object. Validation was performed using test data, separate from the training and validation datasets. Furthermore, a Confusion Matrix was used to assess the performance of this model. The Confusion Matrix is the result of the actual classification prediction and classification in the form of a matrix, which is then used to calculate the values of precision and recall. The formula is as follows:

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

The formula above is calculated with True Positive (TP) being positive cases that are correctly identified, False Positive (FP) being negative cases that were incorrectly classified as positive, and False Negative (FN) as positive cases that were incorrectly classified as negative.

2.4 Application of bytetrack and supervision algorithm

The ByteTrack algorithm is available at https://github.com/ifzhang/ByteTrack. Best model from the previous training, with extension .pt was used as the detection model in the

ByteTrack algorithm. ByteTrack is a tracking model that considers bounding boxes with a low accuracy. ByteTrack works by using a motion model that manages a queue called tracklets to store tracked objects and performs tracking and matching between bounding boxes with low confidence values [5]. ByteTrack is a multi-object tracking algorithm that aims to estimate the bounding boxes and IDs of objects in a video [15]. Object tracking is applied such that during seed counting, moving objects can be tracked and not counted repeatedly. Therefore, this method is suitable for the tracking of moving objects.

In the object counting stage, the algorithm counts the detected and tracked objects. The algorithm used was Supervision, which was obtained from https://github.com/roboflow/supervision. This algorithm counts objects by dividing the frames into 2 counting zones, namely "in" and "out," where there is a counting line on the left side of the frame. This is performed so that objects coming from any direction can be counted when passing through that line. The position of the line is set to pixels (x, y). In this study, the line was located vertically on the left side of the frame, precisely at points (80, 0) to (80, 1072). This stage is applied to three different densities of 20, 40, and 60 eel /0.315 m² as a number variable. This was performed to determine the density at which the model could work optimally.

The tracking and counting stages were evaluated based on the F1 scores. The F1-score was used to determine whether precision and recall values were balanced. According to [8], the F1 score is the result of an accuracy test obtained using the following formula:

$$F1\ score = \frac{2TP}{(TP + FP) + (TP + FN)} \tag{3}$$

In addition, the accuracy of automatic counting (AC) was calculated by comparing the difference with manual counting (MC), referring to [1]. where is the automatic calculation accuracy formula.

$$AC\ accuration = 100\% - \left(\frac{difference}{MC} \times 100\%\right) \tag{4}$$

3 Results and discussion

3.1 Training result

In the training process with the YOLO v8 algorithm, loss graphs and metrics graphs (mAP@0.50, mAP50, mAP@0.50:0.95, or mAP50-95) were generated. Training for both models typically takes approximately 5 h, utilizing two computing units per hour. The compute unit refers to the core clusters on a GPU containing processing elements [10]. The training process of the model can be observed from the graphs in **Fig. 7** and **8**.

Loss represents the sum of errors in the model. This value measures the performance of a model [9]. In **Fig. 7** and **8**, the loss graph tends to decrease towards the x-axis as the number of epochs increases. However, in **Fig. 8**, it can be seen that the head-labelled model tends to have a loss graph closer to the x-axis or number 0 compared to the tail-labelled model. This means that, with the same number of datasets and epochs, the head-labelled model has better learning performance than the tail-labelled model.

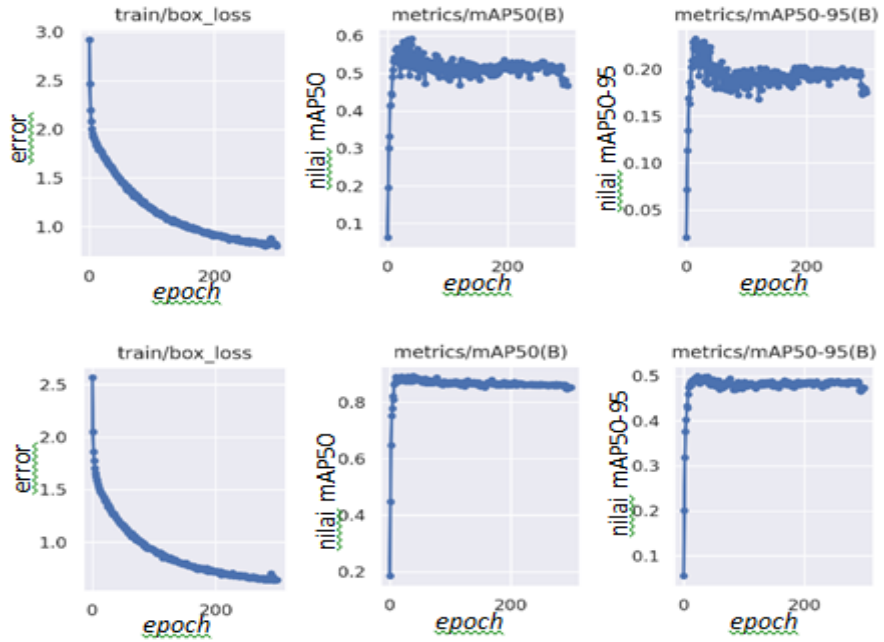


Fig 7. Training loss and metrics graph for tail-labelled model.

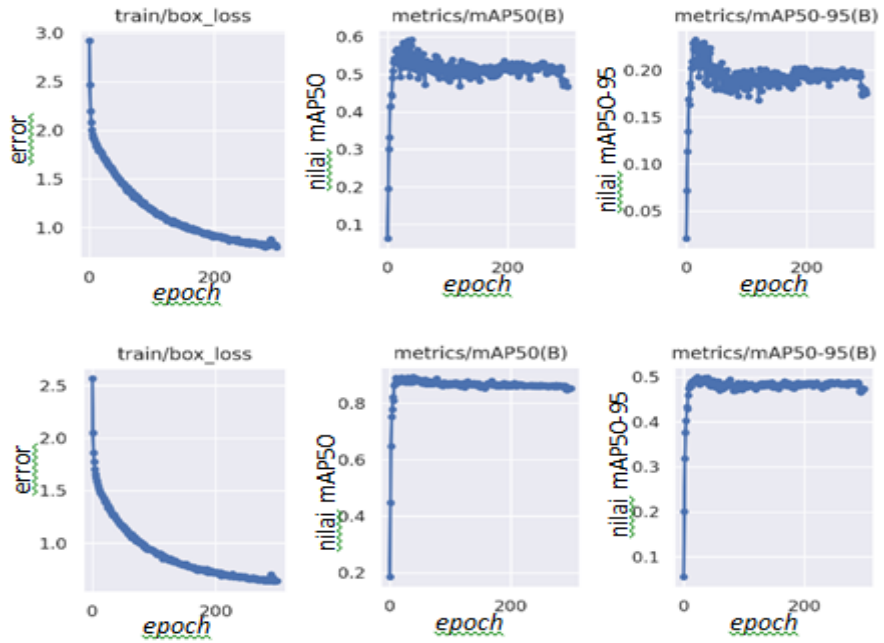


Fig. 8. Training loss and metrics graph for head-labelled model.

In addition to the loss graph, the performance of the model can also be observed in the mAP50 and mAP50-95 metric graphs. In **Fig. 7**, it is evident that the tail-labelled model has a more fluctuating metric graph than the head-labelled model, which tends to be stable. The metrics graph of the head-labelled model also has higher values with mAP50 < 0.90 and mAP50-95 < 0.50 compared to the tail-labelled model with mAP50 < 0.60 and mAP50-95 <

0.25. This indicates that the head-labelled model has a higher level of accuracy in predicting objects than the tail-labelled model.

Model evaluation was performed manually on a test dataset of 10 images. A comparison of the accuracy of the two models, represented by the precision and recall values, is shown in **Tables 1** and **2**.

Table 1. The result of model evaluation on tail-labelled model.

Number of Dataset	TP	FP	FN	Recall	Precision
1	3	1	6	0.33	0.75
2	6	1	9	0.40	0.86
3	5	1	6	0.45	0.83
4	6	2	3	0.67	0.75
5	4	1	6	0.40	0.80
6	6	4	2	0.75	0.60
7	6	2	4	0.60	0.75
8	5	5	7	0.42	0.50
9	8	9	5	0.61	0.47
10	8	9	3	0.73	0.47
Minimum				**0.33**	**0.47**
Maximum				**0.75**	**0.86**
Average				**0.54**	**0.68**

Table 2. The result of evaluation on head-labelled model.

Number of Dataset	TP	FP	FN	Recall	Precision
1	10	0	0	1	1
2	11	0	1	0.92	1
3	8	1	1	0.89	0.89
4	12	0	0	1	1
5	12	1	0	1	0.92
6	11	2	1	0.92	0.85
7	10	3	1	0.91	0.77
8	11	0	3	0.79	1
9	10	0	1	0.91	1
10	10	4	0	1	0.71
Minimum				**0.79**	**0.71**
Maximum				**1**	**1**
Average				**0.93**	**0.91**

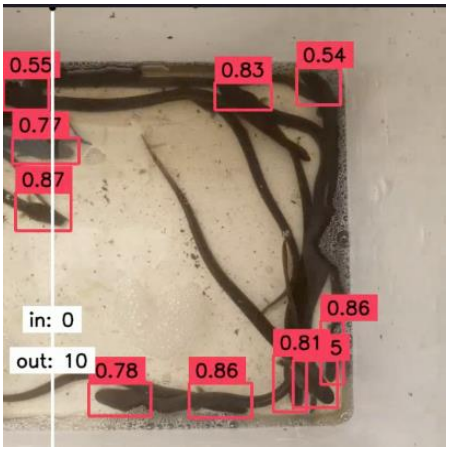
Precision and recall values were obtained from two models: a tail-labelled model and a head-labelled model. The tail-labelled model has an average precision of 0.54 and an average recall of 0.68, while the head-labelled model has higher precision and recall values of 0.93 and 0.91 respectively. The tail-labelled model can achieve a precision and recall value of 1, indicating that it can accurately detect all objects in a frame. However, the head-labelled model is more accurate overall, as it has higher precision and recall values than the tail-labelled model. This suggests that the head-labelled model is less likely to detect false objects or miss real objects.

Object detection using a deep learning method relies on visual characteristics, particularly the colour and shape of body parts. The head of the eel seedling has a distinctive shape, with pectoral fins that resemble ears, whereas the tail does not have unique characteristics and can be mistaken for other objects. In group settings, the tail is often covered by other individuals, whereas the head is more likely to be visible, especially under low water conditions and high density. The ability of eels to extract oxygen from the air is also mentioned as a factor [16].

3.2 Application of bytetrack and supervision algorithm

A previously trained head-labelled model was selected as the detection model for tracking and counting eel seedlings. Object tracking is necessary in this case, because the model is tested on moving objects and cameras. The moving camera adjusted the size of the container according to the ideal frame distance, which was ±60 cm. Object tracking plays a role in tracking eel seedlings by assigning a specific ID such that repeated counting of eel seedlings does not occur. Object detection informs only an object detected at a certain position (x, y) in the frame. Therefore, when the object changes its position, the model considers it as a new and different object from that detected earlier. Object tracking helps the model track the detected object, regardless of its position, as the same object.

The tracked objects can then be counted using object counting. The tracked objects can be counted when the bounding box surrounding the object fully passes the counting line in the frame. The actual number of tested eel seedlings in a 0.315 m² container was 20, 40, and 60. The numbers are derived from the stocking density commonly used by PT. Laju Banyu Semesta hatchery when placing the seedlings in the container, which was 60 seedlings/0.315 m², with a gradual reduction of 20 seeds to make the density difference significantly visible. The results of the object tracking and counting from three different density levels are shown in **Fig. 9**, **10**, and **11**.



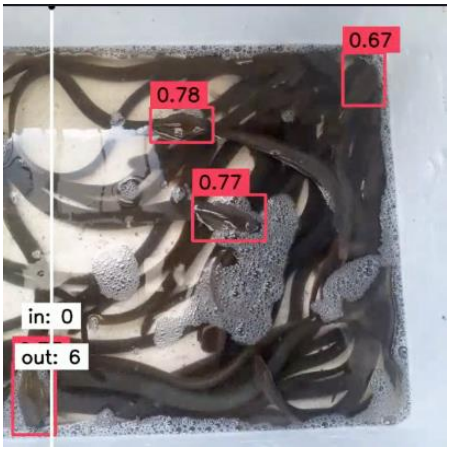


Fig. 9. The result of object tracking and counting on a low density.

Fig. 10. The result of object tracking and counting on a medium density.

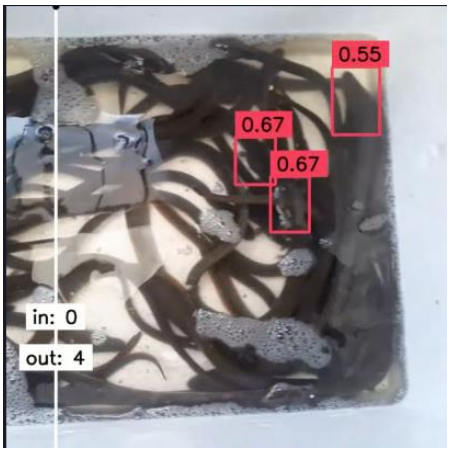


Fig. 11. The result of object tracking and counting on a high density.

Model evaluation was conducted by comparing manual and automatic counts, as well as the F1 score for different densities. Manual counting (MC) was performed on the eel seedlings that crossed the counting line, which consisted of the number of eel seedlings that were correctly detected automatically (AC) and the number of eel seedlings that were not detected automatically (FN). Therefore, for the number of eel seedlings that did not cross the counting line, there were nine, 22, and 41 individuals for each density type. This occurred because, during the video capture, the camera missed capturing the right side of the container, where the camera should have moved to the right until the container with eel seedlings had passed. The results of the manual and automatic calculations of the number of eel seedlings are presented in **Table 3.**

Table 3. The result and evalution of eel seedlings' calculation.

Density	The amount of eel /0.315 m²	AC	MC	TP	FP	FN	F1 Score	Accuracy (%)
Low	20	10	11	10	0	1	0.95	90.91
Medium	40	6	18	6	0	12	0.25	33.33
High	60	4	19	4	0	15	0.17	21.05

Table 3 lists the performance of the model when applied to different density types. At low density, the model achieved an F1 score of 0.95 and a counting accuracy of 90.91%. However, the F1 score and counting accuracy decreased dramatically in the other two densities, namely medium (0.25 and 33.33%) and high (0.17 and 21.05%). This means that the object tracking and counting model is better suited for low-density conditions, specifically 20 individuals/0.315 m² or 63 individuals/m², respectively. Higher densities present factors that reduce detection performance, including the body parts being covered by other individuals and the presence of bubbles owing to the limited oxygen supply.

Object tracking works by tracking the movement of bounding box detection in each frame, also known as tracklets. The proximity between these tracklets determines the object

tracking. Traditional object tracking algorithms typically eliminate bounding box detections with low confidence levels. In contrast, the ByteTrack algorithm takes almost all bounding boxes and separates them into low and high confidence levels, associating them with tracklets. Some tracklets may not match bounding boxes with high confidence levels, which is usually owing to occlusion (partial/complete object obstruction), motion blur, or size changes [15]. In this case, the high density results in increased cases of object fusion, presenting mismatched images in the tracklets. Tracklets with mismatched images were eliminated when compared with tracklets with higher confidence levels. This can be observed in **Fig 12**, where the bounding box detection of eel seedling A slowly disappears because it is attached to eel seedling B.

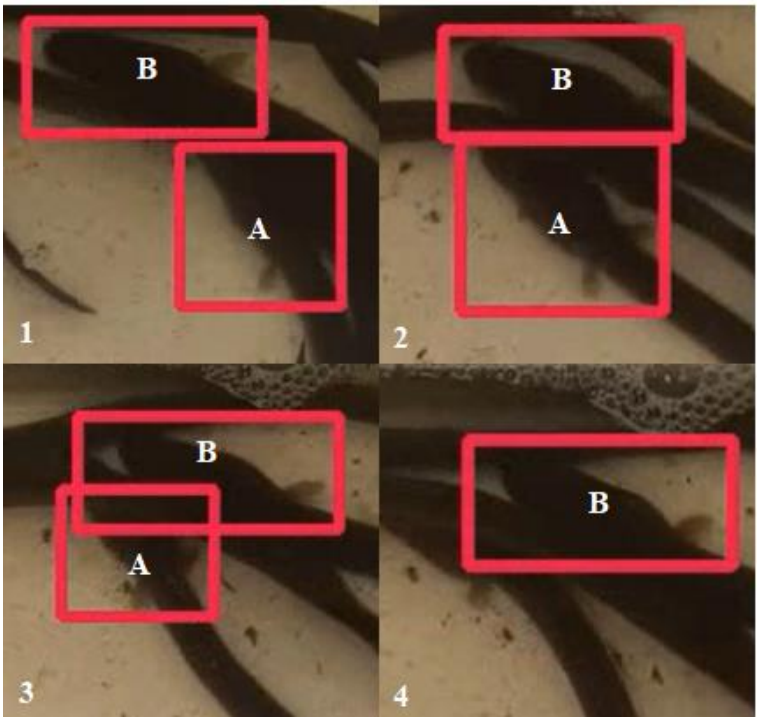


Fig. 12. Object tracking on different frames.

Fig. 12 shows the process of tracking eel seedlings A and B in different frames. In frames 1, 2, and 3, eel seedling A could be seen moving towards eel seedling B and still being tracked as eel seedling A. However, in frame 4, eel seedling A was no longer tracked because the tracklets that previously belonged to eel seedling A were associated with tracklets of eel seedling B. The associated tracklets categorize it as an object and label it as eel seedling B because it has a higher level of confidence.

4 Conclusion

Detection models with head labels have a better detection performance than models with tail labels. The model has precision and recall scores of 0.91 and 0.93, respectively. When applied to object tracking and counting, the model achieved the highest accuracy and F1 score in low-density stocking (63 individuals/m^2), reaching 90.91% and 0.95, respectively.

References

1. Y.P. Admaja, H. Pratikno, W.I. Kusumawati, JoTI. **3**, 1 (2021)
2. N. Afiyat, M. Rifqi, Sim. J. Tek. Mesin. **11**, 1 (2020)
3. F. Aldoni, R. Mukhaiya, J. Multi. RnD. **4,** 2 (2022)
4. G. Chandan, A. Jain, H. Jain, Mohana, ICIRCA. (2018)
5. D. Cochard. *ByteTrack : Tracking model that also considers low accuracy bounding boxes.* https://medium.com/axinc-ai/bytetrack-tracking-model-that-also-considers-low-accuracy-bounding-boxes-17f5ed70e00c. (2021)
6. L.A.F. Fadilla, I.W. Arthana, N.L.A.G. Astriani, G.R.A. Kartika., J. Bumi Lestari. **22**, 2 (2022)
7. L.E. Hadie, E. Kusnendar, Kusdiarti, J Kebijakan Perikanan Indonesia. **13**, 2 (2021)
8. H.V.A. Kautsar, K. Adi, Youngster Physics J. **5**, 1 (2016)
9. A.R. Muhammad, H.P. Utomo, P. Hidayatullah, N. Syakrani, J. Info. Sys. Eng. Business Intelligence. **8**, 1 (2022)
10. F. Murtaza. *AMD Compute Units vs. Nvidia CUDA Cores: What's the Difference?*. https://www.makeuseof.com/compute-units-vs-cuda-cores-whats-the-difference/. (2021)
11. N. Rochmawati, H.B. Hidayati, Y. Yamasari, H.P.A. Tjahyaningtijas, W. Yustanti, A. Prihanto, J. Info. Eng. Edu. Tech. **5**, 2 (2021)
12. D. Nurhayati, S. Hastuti, S.A. Dwiastuti. J. Sains Akua. Trop. **6, 1** (2022).
13. S. Rahmawati, Hasim, Mulis. J. Ilmiah Perikanan dan Kelautan. **3, 2** (2015).
14. A. Sudaryono, S.P. Putro, Suminto. Aquacultura Indonesiana. **15, 1** (2014).
15. Y. Zhang, P. Sun, Y. Jiang, D. Yu, F. Weng, Z. Yuan, P. Luo, W. Liu, X. Wang, Lect. Notes in Computer Science. **13682.** (2022)
16. E. Harianto, E. Supriyono, T. Budiardi, R. Affandi, Y. Hadiroseyani. J. Iktiologi Indonesia. **20, 2** (2019).