ICES Journal of Marine Science

ICES Journal of Marine Science (2021), 78(1), 25–35. doi:10.1093/icesjms/fsaa158

Original Article

Deep learning methods applied to electronic monitoring data: automated catch event detection for longline fishing

Maoying Qiao[1,2], Dadong Wang[1], Geoffrey N. Tuck [2]*, L. Richard Little [2], Andre E. Punt[2,3], and Mike Gerner[4]

[1]*Data61, CSIRO, Sydney, NSW, Australia*
[2]*Oceans and Atmosphere, CSIRO, Hobart, TAS, Australia*
[3]*School of Aquatic and Fishery Sciences, University of Washington, Seattle, WA 98195, USA*
[4]*The Australian Fisheries Management Authority, Canberra, ACT, Australia*

**Corresponding author: tel: +61 0362325222; e-mail: geoff.tuck@csiro.au.*

Qiao, M., Wang, D., Tuck, G. N., Little, L. R., Punt, A. E., and Gerner, M. Deep learning methods applied to electronic monitoring data: automated catch event detection for longline fishing. – ICES Journal of Marine Science, 78: 25–35.

Received 28 August 2019; revised 18 July 2020; accepted 1 August 2020; advance access publication 27 December 2020..

Electronic monitoring (EM) systems have become functional and cost-effective tools for the conservation and sustainable harvesting of marine resources. EM is an alternative to on-board observers, which produces video segments that can subsequently be reviewed by analysts. It is currently used in a range of fisheries. There are two major challenges to the widespread adoption of EM. One is the large storage requirement for the video footage recorded and the other is the long time required by analysts to review the video footage. We propose an automated catch event detection framework to address these challenges. Our solution, based on deep learning techniques, automatically extracts video segments of catch events, which substantially reduces storage space and review time by analysts. Here, we demonstrate the framework using video footage from three longline fishing trips. The system recalled nearly 100% of the catch events across all trips.

Keywords: artificial intelligence, artificial neural networks, deep learning, fisheries management, machine learning

Introduction

Electronic monitoring (EM) systems using on-board cameras are increasingly being used to observe fisheries operations, and in some cases have replaced human observers on vessels. EM systems usually consist of multiple closed-circuit television cameras, a global positioning system (GPS) receiver, a hydraulic pressure sensor indicating fishing start times, a winch sensor, and a system control box (McElderry, 2008), and are deployed for fisheries management purposes to monitor factors such as location, catch, bycatch (Fitzgerald *et al.*, 2019), catch handling, fishing methods, and operational compliance. EM has several advantages in comparison to on-board observers. First, video footage can be kept as a permanent data record for auditing purposes. Second, EM is safer, as there is no need for observers to be on vessels. Third, EM is flexible in terms of scheduling.

While largely in its operational infancy, EM is used in several fisheries worldwide (Helmond *et al.*, 2020). For example, it has been recently fully implemented for the ANABAC-OPAGAC Tropical tuna purse seine programme (Michelin *et al.*, 2018, Helmond *et al.*, 2020), and on over 100 mainly longline vessels, in Australia and the Western Central Pacific (Hosken *et al.*, 2017; 2018).

Despite its increasing use, EM faces considerable challenges associated with storing and analysing the large volume of video data captured. A single fishing trip can generate over 2TB of video data (Gerner, 2015). Analysing such large volumes of video

footage can be labour-intensive, tedious, error-prone, subjective, and costly. Most fisheries using EM only analyse a sample of the data collected, and thus do not fully utilize the large amounts of data captured (Wallace *et al.*, 2015). For example, in Australia's federally managed fisheries, a minimum of 10% of the video footage is reviewed (AFMA, 2019; Gerner, 2015).

Recent developments in artificial intelligence (AI) technologies, particularly deep learning and its application in computer vision, provide possible solutions to these challenges (Malde *et al.*, 2019; Probst, 2019). For example, Bradley *et al.* (2019) proposed a fishery-dependent data collection system to reduce time lags from data collection to action (e.g. management intervention) using automation based on AI. Classical machine learning and computer vision-based techniques, two branches of AI, have been applied to underwater video imagery (Shafait *et al.*, 2016). Many image processing techniques have been used for fish detection and identification (Spampinato *et al.*, 2008), and to measure fish abundance, size structures, species composition, and behaviour (Chuang, 2015). Recent advances in deep learning have also attracted attention. For example, Salman *et al.* (2020) combined raw images from underwater video with Gaussian mixture models and optical flow motion features as input to a deep convolutional neural network (CNN) for fish detection, while Richards *et al.* (2019) presented an open-source platform for underwater image and video analytics using Regions with CNN (R-CNN). French *et al.* (2019) illustrated a system that automatically achieves fish segmentation and species identification on trawler conveyer belts using Mask R-CNN.

Uptake and application of on-board cameras have also kept pace in detecting and measuring on-board landed fish, but only in strictly controlled environments. For example, Miranda and Romero (2017) were able to detect fish but with a fixed background. Similarly, Svellingen *et al.* (2006) utilized a conveyor belt to present fish to the vision system, which was also in a controlled environment. Dual cameras have been used to take stereoscopic images to characterize and measure landed fish (Wallace *et al.*, 2015). These systems tend to require specialized equipment. By contrast, our focus is to analyse video footage captured using operational cameras currently installed on longline fishing vessels. This is challenging because the quality of video footage is often low due to factors such as illumination in rough seas (day and night, sunny, cloudy and stormy weather), uncontrolled background (dry and wet decks), specular reflection and multiple occlusions caused by walking crew members. Images can also be obscured by water droplets that have adhered to the camera lens.

Here, we explore the ability of machine learning to condense the EM video (Figure 1). We use deep learning techniques to identify and store on-board video segments from longline vessels that have "catch events", namely video segments that have at least one fish and one fisher simultaneously on deck. This reduces data storage requirements and increases the percentage of the video containing activities of interest to fisheries managers, and thus improves the efficiency of the data analysis.

Material and methods

We took a two-step approach—first detecting fish and fishers in a frame-by-frame manner, and then smoothing out occasional false and missing detections—to identify catch events and filter out irrelevant or "empty" video segments.

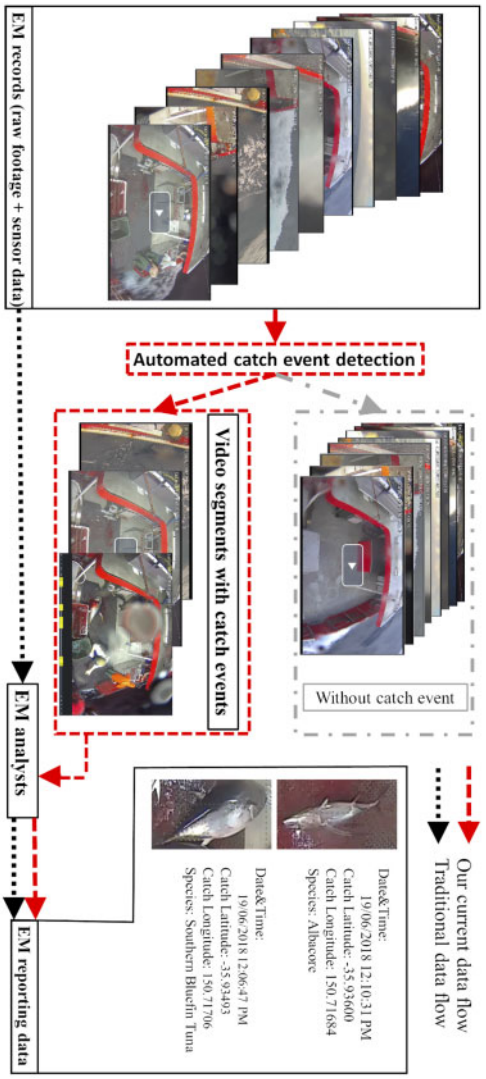

Figure 1. EM data flow. The dotted arrows illustrate the traditional EM data flow; all raw video footage directly go to the EM analysts, and a small portion of the footage is reviewed and used to generate EM reports. The dashed arrows show the EM data flow in our proposed system. Our system filters out the video frames without catch events (gray dot-dashed), and only the video frames with catch events are sent to the EM analysts for review, which aims to improve the efficiency of the analysis.

Data set

We used 140 video segments, in MP4 format, with a resolution of 1280×720 and a framerate of 10, provided by the Australian Fisheries Management Authority. These segments were captured by a camera covering the deck area of a commercial fishing vessel over three trips (85, 31 and 24 segments came from each fishing trip). Each segment lasted either 30 or 60 min, and was viewed and annotated by an EM analyst. The annotation contained details from each fish capture including date and time, location, and species.

Two data sets were constructed. The first was used to train and test a detector of fish and fishers in frames. If a video segment contained continuous frames of simultaneous instances of a fish and a fisher, a catch event was deemed to have occurred. The

second data set was used exclusively for evaluating catch events in video segments.

The first data set was constructed in three steps. First, from the video segments of a single fishing trip, 33 of 85 video segments were identified as having a catch event, i.e. where at least one fish and one fisher were seen simultaneously on the deck of the vessel for a certain amount of time. These were categorized according to the species caught and whether it was day or night. Second, 450 frames that covered a range of species (Figure 2a), fisher stances (Figure 2b), and light conditions were extracted from the video segments. The light conditions included variation in weather, time of day, reflection from the sea surface, and opacity caused by water droplets on the camera lens (Figure 2). Finally, we localized and labelled individual fish and fishers in each frame with tightly bound polygons (Figure 2c). Bounding boxes encompassing the polygons could be extracted for different training requirements. The frames in this data set were subdivided for training (270 frames), validating (to choose the best model; 90 frames) and testing (90 frames; Table 1).

The second data set had three components. The first component contained the 85 video segments from the first data set. Segments having at least one catch event were labelled with 1; the others were labelled with 0. The second component contained the 33 video segments containing at least one catch event, indicated previously, but with the start and end times of each catch event labelled, to evaluate how accurately the system could locate catch events. The number of catch events per video segment is summarized by the black bars in Figure 3a. Most of the video segments (20 out of 33) had more than one catch event. Most catch events lasted between 1 and 10 min (black bars, Figure 3b), depending on the time taken by a fisher to process a fish. A typical process included hauling a fish, dragging the fish, handling the fish, wrapping the fish, and storing the fish. The third component was

Table 1. The number of frames and the number of instances of fish and fishers in the frames used for training, testing and validating fish and fisher detection.

Categories	Purpose	Frames, *n* (%)	Instances
Fish	Training	270 (60)	647
	Testing	90 (20)	227
	Validating	90 (20)	183
Fisher	Training	270 (60)	988
	Testing	90 (20)	347
	Validating	90 (20)	294

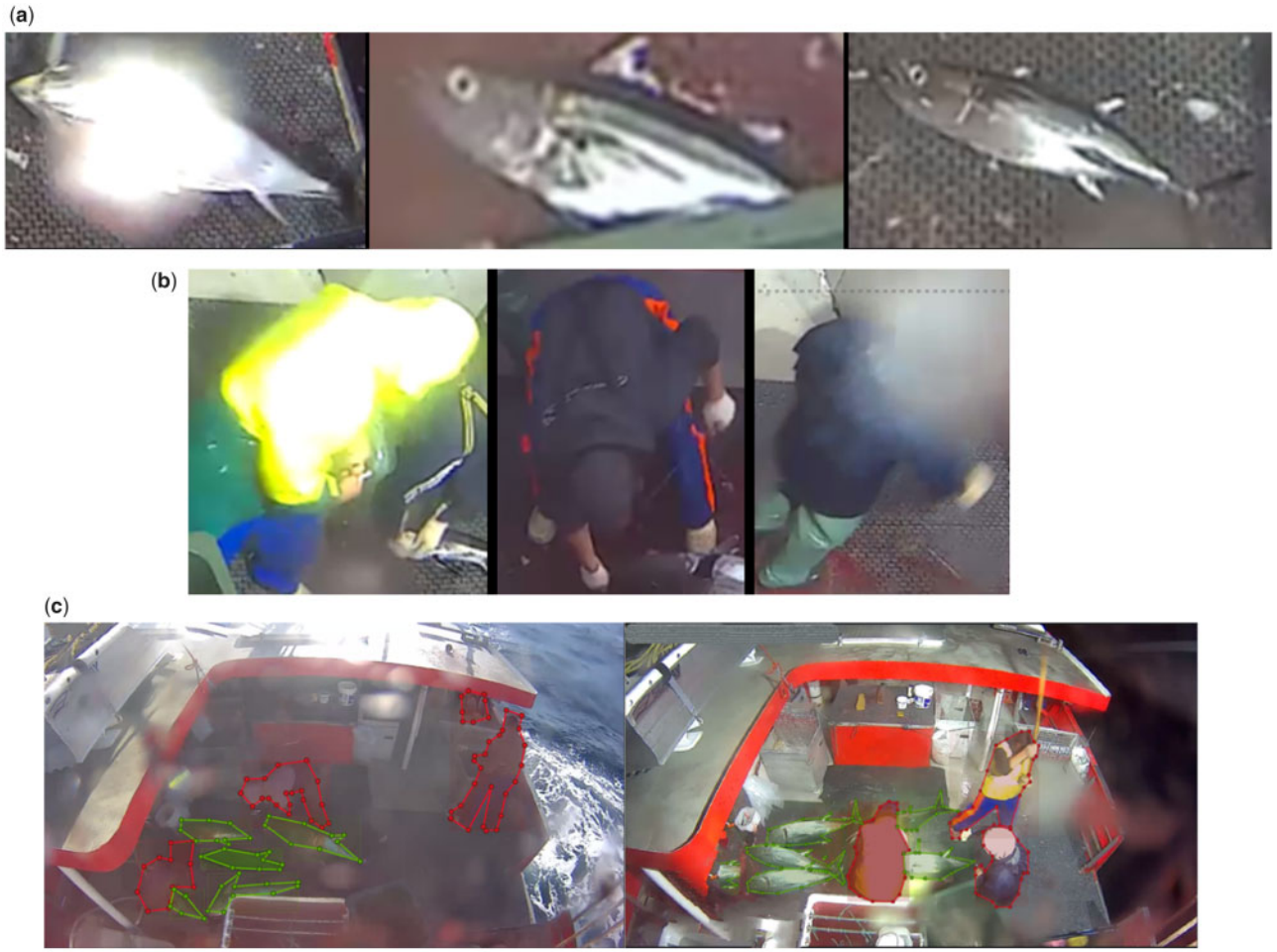

Figure 2. Example frames from our data sets. (a) Fish appearance variation is due to fish species (e.g. albacore, southern bluefin tuna, swordfish, ocean sunfishes, blue shark, and lancetfishes), fish orientation, light reflection, occlusion, etc. (b) Human appearance variation includes poses, occlusion, light condition, and blurriness. (c) Example frames captured from the longline vessel. Our target objects, i.e. fish and fisher, are labelled with polygons. The left frame illustrates a scene from day time, and the right was from night time.

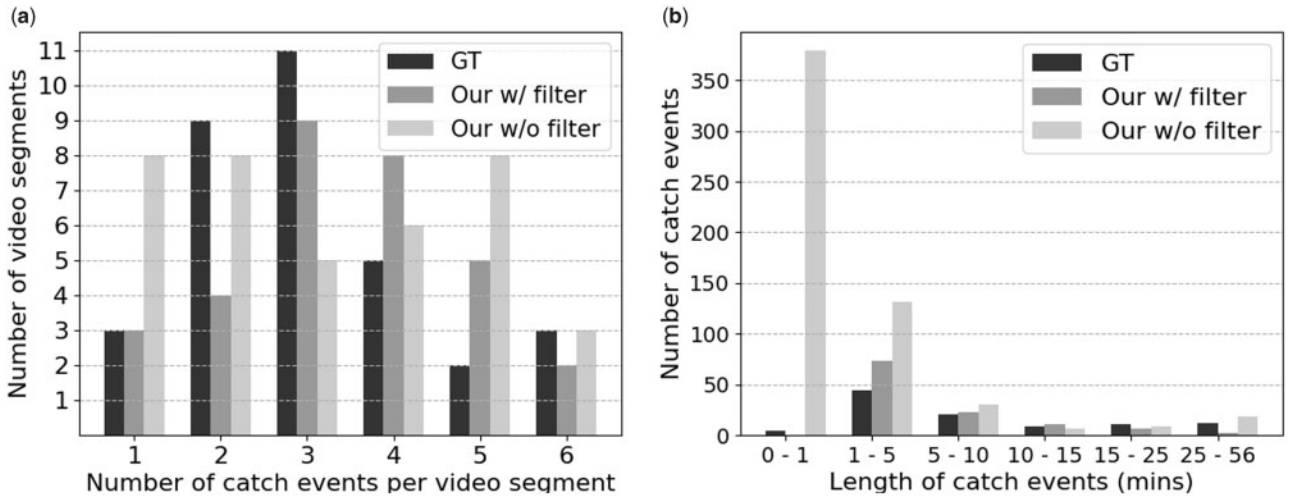

Figure 3. Statistics of catch events from different methods: GT—the ground-truth number in the data set, Our w/filter—the number counted using our system, and Our w/o filter—the number counted from the prediction that were not smoothed by the temporal filter. (a) Histogram of the number of catch events per video segment. There are in total 102 catch events from the ground-truth, 116 from the prediction with the proposed temporal filter, and 581 from the prediction without the temporal filter. (b) Histogram of the lengths of the catch events. The specific histogram numbers were (5, 44, 21, 9, 11, 12) for ground-truth, (0, 73, 23, 11, 7, 2) for predictions with the temporal filter, and (379, 132, 30, 7, 9, 18) for predictions without the temporal filter.

constructed from the other two trips to evaluate the broader applicability of the detector. Similar to the second component, the start and end times of all catch events in the 31 and 24 video segments of these two trips were labelled.

The deep learning system

The deep learning system consisted of two parts: an object detection framework based on CNN, and a catch event detection framework based on a temporal filter (Figure 4a).

Object detection

Deep learning based multi-class object detection frameworks include two-stage methods such as R-CNN (Girshick *et al.*, 2016), Fast and Faster R-CNNs (Ren *et al.*, 2015; Wang *et al.*, 2017), SPPNet (He *et al.*, 2015), and RFCN (Dai *et al.*, 2016), and one-stage methods such as YOLO/YOLOv2/YOLO9000 (Redmon *et al.*, 2016; Redmon and Farhadi, 2017) and single shot detector (SSD) (Liu *et al.*, 2016). We experimented with two frameworks, TensorBox (based on its success for people detection; Stewart *et al.*, 2016; Stewart, 2016) and YOLO (due to its real-time processing capacity), for fish and fisher detection.

Both of the object detection frameworks are composed of convolutional layers for feature extraction and fully connected layers for bounding box regression and target object classification. The CNN of TensorBox was implemented using GoogLeNet (Szegedy *et al.*, 2015). Its framework is demonstrated in Figure 5. We also examined other CNN architectures that worked effectively for feature extraction in classification tasks, such as ResNet (Szegedy *et al.*, 2017) and DenseNet (Huang *et al.*, 2017), to identify the architecture that is best suited for our task. These CNN architectures have different convolutional calculations for feature extraction but share the input layer and the fully connected layer (Figure 5). The CNN of YOLO was implemented in Darknet (https://pjreddie.com/darknet/yolo/). The main differences among these CNN architectures relate to the basic blocks on which they are built. ResNet is based on a shortcut connection with residual learning and enables training extremely deep neural networks (Szegedy *et al.*, 2017). GoogLeNet (Szegedy *et al.*, 2015) is based on inception modules, which allow for more units at each stage without an uncontrolled increase in computational complexity. DenseNet designs shorter connections between layers close to the input and layers close to the output for neural networks that are deeper, more accurate and efficient to train. The architecture in YOLO was inspired by GoogLeNet but with simpler building blocks than the inception modules.

The objective loss ℓ of the fully connected layers is composed of two parts (Girshick *et al.*, 2016): one for localization loss and one for object classification loss. Formally, the multi-task objective loss is:

$$\ell = \alpha_{\text{reg}}\ell_{\text{reg}} + \alpha_{\text{CE}}\ell_{\text{CE}}, \tag{1}$$

with ℓ_{reg} representing the bounding box regression loss and ℓ_{CE} the cross-entropy classification loss. The two parameters α_{reg} and α_{CE} are trade-off parameters and are used to balance the learning between regression and classification.

For the bounding box regression task, the loss is defined as:

$$\ell_{\text{reg}} = \frac{1}{N_{\text{pos}}}\sum_i \mathbb{1}_i^g\left(|x_i - x_i^g| + |y_i - y_i^g| + |w_i - w_i^g| + |h_i - h_i^g|\right), \tag{2}$$

where $(x_i,\ y_i,\ w_i,\ h_i)$ represent the centre coordinates, and width and height of the i-th predicted bounding box. Symbols with a g superscript represent data (i.e. labelled or ground-truthed) to which the detector is being fitted. $\mathbb{1}$ filters the predictions that have overlap with ground truth and are labelled as "positive" for the classification task. N_{pos} represents the total number in the category. The other predicted bounding boxes are labelled as "negative". Only positive predictions contribute to the regression loss in (2).

For the classification task, the cross-entropy loss is applied, defined as

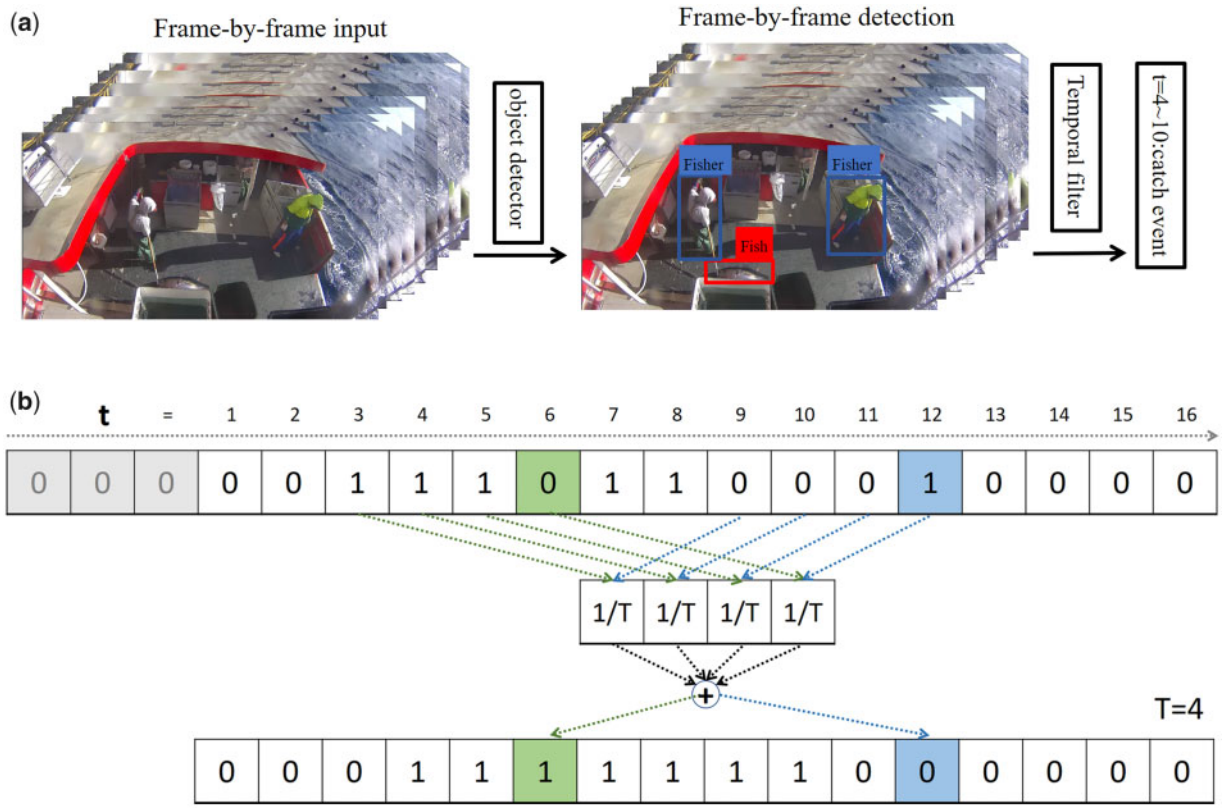

Figure 4. The deep learning system applied to fisheries electronic monitoring. (a) The catch event detection framework: The CNN-based object detector takes video as input and conducts frame-by-frame object detection. The blue bounding boxes indicate detected fishers and the red ones indicate detected fish. Based on those predictions, a temporal filter is utilized to filter out video segments without catch events, and only the video segments with catch events are kept. (b) The mechanism of the temporal filter: A binary sequence corresponding to a video segment with 16 frames is taken as an example here, as shown in the top vector. Its entries with binary values are converted from the frame-by-frame object detections. Value 1 indicates that both fish and fisher are detected in the frame, while 0 indicates all other situations. An averaging convolutional kernel (with length $T = 4$ here, shown in the middle vector) slides along the sequence, and a smoothed binary sequence (shown in the bottom vector) is then obtained. The output 1 highlighted in green is recovered from 0 (a possible missing detection). In contrast, the output 0 highlighted in blue is rectified from 1 by filtering out occasional false positives.

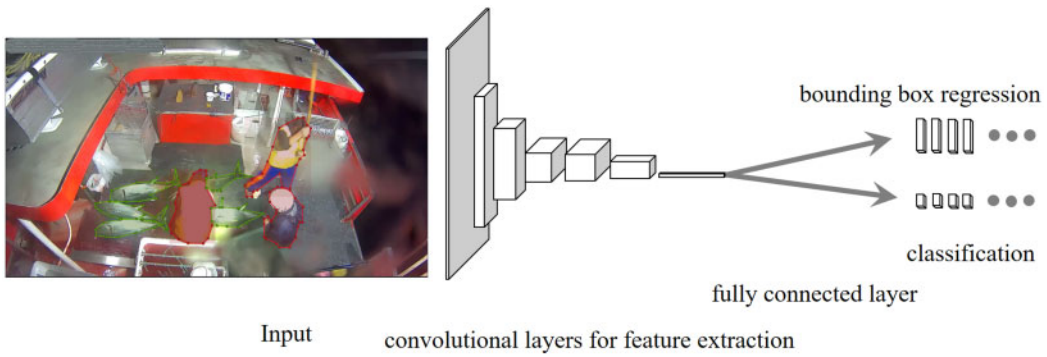

Figure 5. Object detection framework with GoogLeNet as its backbone for feature extraction. Convolutional layers convert an input image with resolution 1280×704 into smaller feature maps to do bounding box regression and classification. To test other architectures, the convolutional layers were replaced with ResNet (Szegedy *et al.*, 2017) and DenseNet (Huang *et al.*, 2017).

$$\ell_{\mathrm{CE}} = -\frac{1}{N_{\mathrm{pos}} + N_{\mathrm{neg}}} \sum_i \left(t_i \log p_i - (1 - t_i) \log(1 - p_i) \right), \qquad (3)$$

where $t_i \in \{0, 1\}$ is the ground-truth label for the i-th predicted bounding box, where 0 indicates "negative" and 1 "positive"; $p_i \in [0, 1]$ is the confidence score assigned by a sigmoid activation function, which is formulated as $p_i = \frac{1}{(1+e^{-wf_i})}$ with f_i the CNN feature vector of the i-th bounding box, and w the vector of feature weights.

Once the detector is trained, the testing data set is used to verify its effectiveness.

Catch event detection

We assume that a catch event has occurred when both fish and fishers simultaneously appear in continuous frames. A temporal filter was designed to automatically detect catch events based on the results of fish and fisher detection. Once all frames had predictions of bounding boxes, they were allocated a sequence of binary values. A frame was allocated a 1 if it contained both fish and fisher predictions, and a 0 otherwise. A one-dimensional averaging filter of length T was then applied (Figure 4b), with T fixed to 300 in our analyses. The filter convolutes over the whole sequence to generate a smoothed sequence of binary values. This results in occasional inconsistencies that were filtered out, as demonstrated in Figure 4b, which enhanced the frame-by-frame detection results. For example, a "0" amongst 1s represents missing detections because both the predicted fish and fishers could not disappear and reappear in a short period of time. Such a frame might be caused by occluded fish. Alternatively, a "1" amongst 0s represents a false positive, which should be ignored. Finally, a catch event is confirmed if the length of a sequence containing constituent 1s is over a threshold L_T. For instance, if $L_T = 7$, then one catch event is detected, which happens say from $t = 4$ to $t = 10$. Otherwise, no catch event is detected in the video segment.

Evaluation metrics

Two metrics were applied to evaluate the accuracy of the object detector: precision and recall (Powers, 2011), which are defined as:

$$\text{Precision} = \frac{\text{true positives}}{\text{true positives} + \text{false positives}},$$

$$\text{Recall} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}}.$$

where a true positive refers to predicted bounding boxes that are matched to ground-truth positives, i.e. ground-truth bounding boxes for fish and fishers, while a false positive refers to the predicted bounding boxes having no overlap with any ground-truth bounding boxes. False negatives refer to the ground-truth bounding boxes missed by the detector. Precision and recall evaluate different aspects of the detector. Precision calculates the percentage of predicted bounding boxes that are relevant, whereas recall measures the percentage of the relevant bounding boxes that have been retrieved relative to the total number of ground-truth bounding boxes. The average precision (AP) score and recall rate are calculated as in Manning *et al.* (2008) to numerically summarize the precision of the detector.

There is one adjustable parameter for the evaluation metrics in the detection framework, i.e. the *Intersection over Union* (IoU) threshold. The IoU represents the proportion of overlap between a predicted object and its corresponding ground-truth. For evaluation purposes, successful detections were determined to be true if the IoU was above an IoU threshold. We present results for three IoU thresholds, i.e. (0.2, 0.5, and 0.7).

A confusion matrix (Stehman, 1997) was used to evaluate the performance of the catch event detector in terms of the binary labelling, i.e. a video segment has or does not have a catch event. In addition, two measures—temporal precision and temporal recall—were developed to evaluate how well the predicted segments

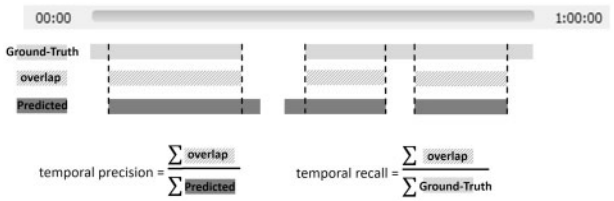

Figure 6. Illustration of the calculation of temporal precision and temporal recall.

overlapped with the ground-truth segments (Figure 6). This involved first calculating the time of overlap between ground-truth catch events and predicted catch events. Temporal precision was then calculated as the ratio of the time of the overlap to the time events that were predicted to occur. Temporal recall was calculated as the ratio of the time of overlap to the time associated with the ground-truth segments.

Experimental specifications

The training data set was augmented as part of data preparation. Per-pixel mean subtraction and greying were applied to emphasize the shape and texture of the targets and weaken the intensity variation. RMSprop, an adaptive learning rate optimizer, was used with the learning rate starting from 0.001, and decreasing by a factor of 0.9 every 5000 iterations. Convergence occurred at 12 000 iterations, approximately equal to 10 epochs. The fully connected layer was initialized with a uniform distribution over the range $(-0.1, 0.1)$. Learning from both random weight initialization and a pre-trained model within the TensorBox framework and a pre-trained model within YOLO for feature extraction were tested. The pre-trained model was obtained using an irrelevant large-scale ILSVRC-2012-CLS image classification data set (https://github.com/tensorflow/models/tree/master/research/slim). Finally, dropout was applied with a probability of 0.5 on the fully connected layer to regularize the network and prevent over-fitting.

We placed a constraint of 1 minute as the minimum time for a catch event given that it involves several steps. Thus, for the catch event detector, the threshold L_T of the temporal filter was fixed to 600 frames (60 seconds). This setting was supported by the ground-truth catch event detection data set from the first trip where 97 out of 102 catch events lasted longer than 1 minute (Figure 3).

Results

Object detection results

TensorBox with CNN architectures study

We evaluated various CNN architectures—ResNet, GoogLeNet, and DenseNet—to identify that most suitable for our task. AP and recall (Tables 2 and 3) decreased with an increasing IoU threshold. Detections using pre-trained models were better than detections using random weight initializations in terms of both AP and recall. Detectors worked better for fishers than fish, because there were fewer training instances of fish but higher appearance variation. Lastly, the CNN architecture ResNet 152 achieved the best performance for both fisher and fish detection for various IoU thresholds.

Based on these observations, we employed the fish and fisher detector with a ResNet 152 CNN and pre-trained initializations

Table 2. Average precision for TensorBox with six CNN architectures, three levels of IoU and two initialization strategies.

		IoU					
		0.2		0.5		0.7	
Architectures	**Initialization**	**Fishers**	**Fish**	**Fishers**	**Fish**	**Fishers**	**Fish**
ResNet 50	Pre-trained	**0.989**	**0.997**	**0.984**	0.967	0.862	0.700
	Random	0.960	0.994	0.938	0.959	0.735	0.645
ResNet 101	Pre-trained	**0.987**	**0.997**	0.982	**0.979**	0.862	**0.740**
	Random	0.931	0.995	0.905	0.959	0.661	0.664
ResNet 152	Pre-trained	**0.987**	**0.998**	**0.984**	**0.983**	**0.922**	0.732
	Random	0.960	0.994	0.938	0.959	0.735	0.645
GoogLeNet	Pre-trained	0.977	0.990	0.961	0.942	0.786	0.669
	Random	0.977	0.996	0.948	0.947	0.606	0.706
DenseNet121	Pre-trained	0.985	0.994	0.973	0.977	0.815	**0.853**
	Random	0.972	**0.998**	0.930	0.931	0.672	0.482
DenseNet 169	Pre-trained	0.984	0.995	0.972	0.965	**0.873**	0.698
	Random	0.964	0.985	0.927	0.968	0.632	0.743

Red font highlights the best performance and bold highlights the second-best performance. The grey highlighted row shows the best model in terms of overall performance.

Table 3. Recall for TensorBox with six CNN architectures, three levels of IoU and two initialization strategies.

		IoU					
		0.2		0.5		0.7	
Architectures	**Initialization**	**Fishers**	**Fish**	**Fishers**	**Fish**	**Fishers**	**Fish**
ResNet 50	Pre-trained	**0.824**	**0.863**	0.804	0.775	0.611	0.480
	Random	0.620	0.758	0.568	0.670	0.320	0.326
ResNet 101	Pre-trained	0.807	0.859	0.784	0.797	**0.643**	0.493
	Random	0.689	0.775	0.643	0.665	0.398	0.379
ResNet 152	Pre-trained	0.818	0.841	**0.804**	**0.797**	**0.671**	**0.542**
	Random	0.620	0.758	0.568	0.670	0.320	0.326
GoogLeNet	Pre-trained	0.816	**0.890**	0.775	**0.802**	0.516	0.458
	Random	0.801	0.771	0.738	0.652	0.450	0.313
DenseNet121	Pre-trained	**0.841**	0.850	**0.801**	0.789	0.614	**0.546**
	Random	0.784	0.793	0.709	0.617	0.403	0.286
DenseNet 169	Pre-trained	0.816	0.815	0.781	0.736	0.625	0.432
	Random	0.744	0.806	0.686	0.714	0.392	0.383

Red font highlights the best performance and bold highlights the second-best performance. The grey highlighted row shows the best model in terms of overall performance.

for the following catch event detection task. Examples of frame-by-frame object detection results from the testing data set shown in Figure 7 cover considerable variation of footage conditions such as day and night illumination conditions, blurred frames, and light reflection. The object detector identified most of the fishers, which is consistent with the numerical evaluation metrics. Most of the fish with high appearance variations, e.g. scales, shapes, and species, were also identified. In some cases the detector failed to detect fish due to occlusions caused by walking crew members on the vessel deck, and this leads to the low recall rate.

The performance of TensorBox vs. YOLO

YOLO did not perform as well as TensorBox in terms of AP, but obtained better results in terms of recall and was faster (Table 4). The detector worked well on the second and the third trips of the evaluation data set (Figure 8). The similarity of the vessel deck background between the fishing trips is likely the main reason that the detector performed well. The confidence for fishers was not as high as the detections for the trip used for training (Figure 9) because of differences in fisher appearance between trips. For example, the vessel crew wore dark blue and dark green working suits for the trip used for training (Figure 7) but wore bright blue or yellow suits (Figure 8) for the second and the third trips. This could be resolved by adding more training samples to increase the diversity of the training data set.

Catch event detection results

Results on the video segments from the first trip used to extract training frames

The confusion matrix of catch event detection on the data set from the first trip is summarized in Table 5. The catch event detector achieved 100% precision and 97% recall, resulting in an accuracy of 98.8%. Video segments with a predicted "1" label all contained catch events, but not all video segments with

catch events were retrieved. This is mainly due to missed catch events of short duration. The results were promising compared to the approach of random assignment (half of the ground-truth 1s were labelled as 1 and the remaining half were labelled as 0, with the same procedure applied to ground-truth 0s).

The catch event detector also predicted the start and end time of catch events. The number of predicted catch events per video segment was larger than the ground-truth, as indicated by higher numbers in grey bars in Figure 3a. This might be caused by missing fish lying stationary on the deck between fish catch events.

We did not predict catch events shorter than 1 min due to the setting of our temporal filter (grey bars in Figure 3b). In addition, the number of catch events lasting longer than 15 min detected was smaller than the actual number of ground-truth catch events of that duration, while the number of catch events between 1 and 5 min was larger than the actual number of that duration. We attribute this to the missing detection of stationary fish that were lying on the cutting deck for a long time, which split a long catch event into several shorter ones.

To verify the effectiveness of the proposed temporal filter, we evaluated the catch event detector when no temporal filter was applied. Each sequence of consecutive frames containing simultaneous fish and fishers was deemed as one catch event. The light grey bars in Figure 3 show that most of the catch events were short and lasted less than a minute. This is because occasional missing detections separate a long catch event into several shorter ones. This result verifies the value of the temporal filter. Two visual examples demonstrate the effect of the temporal filter further (Figure 9). One is to keep frames without fish or fisher detections within a catch event. The other is to ignore frames with fish and fisher detections outside of catch event segments.

The temporal precision and temporal recall for this trip were 100 and 62.92%, respectively. In other words, all catch events were correctly predicted, but some were missed. This might be caused by missing detections of fish that were lying stationary on the cutting deck between catch events.

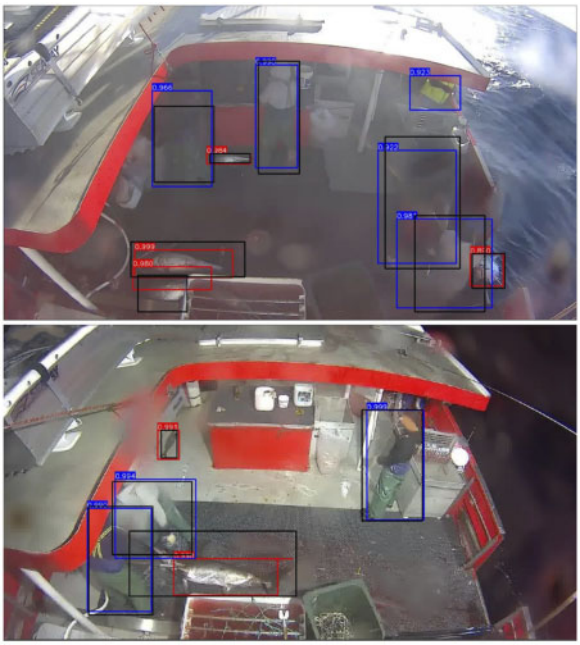

Figure 7. Examples of object detection results. Black bounding boxes indicate ground-truth positions of fishers and fish. Blue bounding boxes represent located fishers using our detector, while red bounding boxes show detected fish. The numbers, along with bounding boxes are the confidence scores of the detected objects (fish or fish).

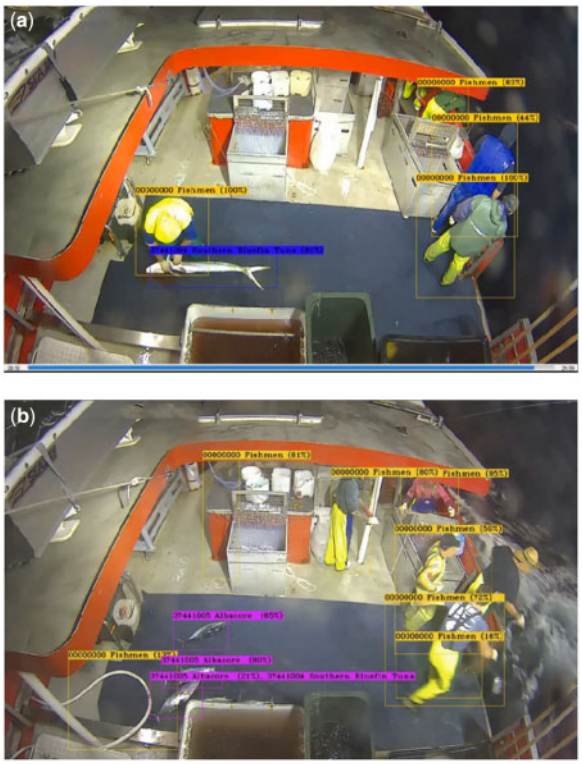

Figure 8. Fish/fisher detection results for frames from the second and third trips in the evaluation data set. (a) The crew members wearing bright blue suits. (b) The crew members of another trip wearing bright yellow suits.

Table 4. Comparison of average precision and recall between YOLO and TensorBox (IoU = 0.2).

Model	Architectures	Initialization	AP		Recall		Prediction time for 1-h video (minutes)
			Fishers	Fish	Fishers	Fish	
TensorBox	ResNet152	Pre-trained	0.987	0.998	0.818	0.841	>60
YOLO	Darknet	Pre-trained	0.89[a]		0.88[a]		≈30

[a]Mean AP.

Results on video segments from the second and third trips

The temporal precision and recall of the video segments with catch events for these two trips (Table 6) show that the catch event detector performed well on trips other than that used for training.

Discussion

A key need for sustainable fisheries management is comprehensive and reliable data. Fisheries independent coverage, usually by human observers, is often limited or non-existent. This is largely due to cost and the remoteness of fishing (Bradley *et al.*, 2019). EM systems have the ability to overcome these constraints. However, broad-scale adoption of EM remains limited due to the costs associated with data storage and viewing. The work proposed here is significant because it will reduce both storage requirements and review time. In time, we envisage the automation of vessel catch and bycatch identification will operate in real-time, allowing immediate logging of catch data and the rapid response by regulators to unwanted interactions with endangered species. This will be a considerable leap forward in the management of fisheries. This paper describes the first steps towards that outcome.

Although EM has been trialled and fully implemented in many fisheries worldwide, automated video analysis systems have not been applied widely (Helmond *et al.*, 2020). In this study, we have developed a deep learning-based catch event detection system to fill this gap. It condenses the raw video footage obtained by EM and thus improves efficiencies of storage and review. To the best of our knowledge, this is the first system to detect catch events from on-vessel EM video footage. From a technical view, we found that deep learning-based object detectors—the core component of our system—worked well for our task even with a small training data set. It even performed better on our data set in terms of AP than on one of the benchmark data sets—MS COCO object detection data set, although they had different IoUs (Bochkovskiy *et al.*, 2020). This may be because our data set has fewer classes. We expect that this promising result will encourage more fish and fisher detection application scenarios, such as fish catch and composition analysis. Tseng and Kuo (2020) recently used a Mask R-CNN for object detection and a segmentation method to prescreen harvested fish. This work produced accurate counts of the harvested fish. By contrast, our intent was to help analysts review EM to generate accurate catch reports efficiently.

While we have used footage from three trips from a single vessel, ideally, the software should be portable across vessels in a fleet. Depending on fleet size, labelling training data and training a detector for every vessel may be feasible because the size of the training data set was not large in our experiments and the training procedure can be automatized. Another potential solution to

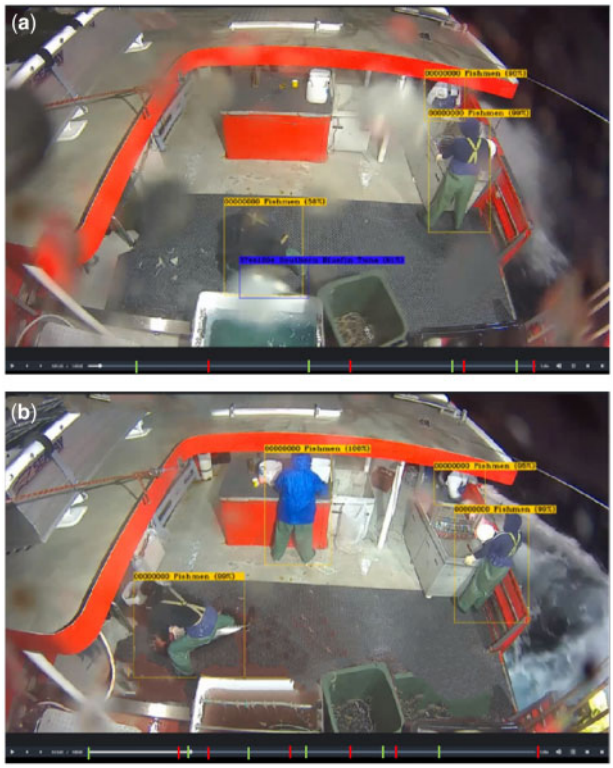

Figure 9. Two snapshots of the catch event detection software, demonstrating the effect of the temporal filter. Indicators, i.e. green bars and red bars in the timelines, show the start and ending time stamps of detected catch events. (a) A frame with a fish detected in a non-catch-event period. This fish detection was filtered out by the temporal filter and was not labelled as a catch event because it occurs outside of the start of the first detected catch event in the video segment. (b) A frame with a missing fish. The human-occluded fish was missed by the object detector. However, after applying the temporal filter, this frame was kept as part of a catch event, as this frame was within green and red bars as shown by the software interface.

Table 6. Temporal precision and temporal recall for the second and third trips.

	Temporal precision (%)	Temporal recall (%)
Second trip	99.93	65.61
Third trip	99.98	62.32

Table 5. Confusion matrix of catch event detection for the data set from the first trip for the evaluation of the performance of the catch event detector, comparing our method with random assignments (in parentheses).

Predicted Ground-truth	1	0	Recall
1	32 (17)	1 (16)	32/(32 + 1) = 0.970 [17/(17 + 16) = 0.515]
0	0 (26)	52 (26)	–
Precision	32/(32 + 0)=1.000 [17/(17 + 26) = 0.395]	–-	(32 + 52)/85 = 0.988 (Accuracy) [(17 + 26)/85 = 0.506]

scalability is a generic fish/fisher detector that could be trained on a large-scale data set. However, more research is needed to determine how many training video frames are needed to achieve satisfactory performance for the whole fleet. Factors that need to be considered when constructing such a data set include, but are not limited to, good coverage of fish species (bycatch of rare species will need to be considered carefully), vessel setups, camera views, weather conditions, as well as day and night illumination variations.

Acknowledgements

The authors would like to thank the support from CSIRO Research Plus Postdoctoral Fellowship, AFMA for providing data and discussing the features of our system, and Logan Howell for assisting the construction of the data set for fish and fisher detection.

Data availability

The data underlying this article were provided by the Australian Fisheries Management Authority (AFMA) by permission. The data underlying this article cannot be shared publicly due to the commercial in confidence nature of the on-vessel footage. Data requests can be made to the Australian Fisheries Management Authority.

References

AFMA. 2019. Australian Fisheries Management Authority Electronic Monitoring Program—Program Overview. https://www.afma.gov.au/sites/g/files/net5531/f/uploads/2014/12/Commonwealth-e-Monitoring-Program_program-overview_August-2015-update.docx (last accessed 25 May 2020).

Bradley, D., Merrifield, M., Miller, K. M., Lomonico, S., Wilson, J. R., and Gleason, M. G. 2019. Opportunities to improve fisheries management through innovative technology and advanced data systems. Fish and Fisheries, 20: 564–583.

Bochkovskiy, A., Wang, C. and Liao, H. 2020. YOLOv4: optimal speed and accuracy of object detection. ArXiv: 2004.10934.

Chuang, M. 2015. Automatic video analysis for fisheries survey systems. Doctoral dissertation, University of Washington, Seattle, WA.

Dai, J., Li, Y., He, K. and Sun J. 2016. R-FCN: Object detection via region-based fully convolutional networks. *In* Proceedings of the Thirtieth Conference on Advances in Neural Information Processing Systems, pp. 379–387, Barcelona, Spain.

Fitzgerald, S., Wallace, F., Romain, S., Magrane, K., Kazmerzak, R., Moore, B., and Kim, M. A. 2019. Improving seabird species identification in electronic monitoring applications using machine learning systems. *In* Ninth Meeting of the Seabird Bycatch Working Group, Florianopolis, Brazil.

French, G., Mackiewicz, M., Fisher, M., Holah, H., Kilburn, R., Campbell, N., and Needle, C. 2019. Deep neural networks for analysis of fisheries surveillance video and automated monitoring of fish discards. ICES Journal of Marine Science, 77: 1340–1353.

Gerner, M. 2015. Cost effective monitoring in Australia's tuna longline fisheries. *In* International Workshop on the Application of Electronic Monitoring Systems in Tuna Longline Fisheries, Kaohsiung City, Taiwan.

Girshick, R., Donahue, J., Darrell, T., and Malik, J. 2016. Region-based convolutional networks for accurate object detection and segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 38: 142–158.

He, K., Zhang, X., Ren, S., and Sun, J. 2015. Spatial pyramid pooling in deep convolutional networks for visual recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, 37: 1904–1916.

Helmond, A. T. M., Mortensen, L. O., Hansen, K. S. P., Ulrich, C., Needle, C. L., Oesterwind, D., Larsen, L. K. *et al.* 2020. Electronic monitoring in fisheries: lessons from global experiences and future opportunities. Fish and Fisheries, 21: 162–189.

Hosken, M., Williams, P., and Smith, N. 2017. A Brief Up-date on ER and EM Progress in the Region. https://www.wcpfc.int/file/144696/ (last accessed 25 August 2020).

Hosken, M., Williams, P., Smith, N., Loganimoce, E., and Schneiter, E. 2018. ER and EM Implementation Progress in the Region. https://www.wcpfc.int/node/31029 (last accessed 25 August 2020).

Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q. 2017. Densely connected convolutional networks. *In* Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4700–4708, Honolulu, Hawaii.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. C. 2016. SSD: single shot multibox detector. *In* Proceedings of the Fourteenth European Conference on Computer Vision, pp. 21–37, Amsterdam, the Netherlands.

Malde, K., Handegard, N. O., Eikvil, L., and Salberg, A.-B. 2019. Machine intelligence and the data-driven future of marine science. ICES Journal of Marine Science, 77: 1274–1285.

Manning, C., Raghavan, P. and Schütze, H. 2008. Introduction to information retrieval. Cambridge University Press, Cambridge.

McElderry, H. 2008. At-sea observing using video-based electronic monitoring. In *Background paper prepared by Archipelago Marine Research Ltd for the Electronic Monitoring Workshop* July, pp. 29–30.

Michelin, M., Elliott, M., Bucher, M., Zimring, M., and Sweeney, M. 2018. Catalysing the Growth of Electronic Monitoring in Fisheries. https://www.nature.org/content/dam/tnc/nature/en/documents/Catalyzing_Growth_of_Electronic_Monitoring_in_Fisheries_9-10-2018.pdf (last accessed 25 August 2020).

Miranda, J. M., and Romero, M. 2017. A prototype to measure rainbow trout's length using image processing. Aquacultural Engineering, 76: 41–49.

Powers, D. M. 2011. Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. Journal of Machine Learning Technologies, 2: 37–63.

Probst, W. N. 2019. How emerging data technologies can increase trust and transparency in fisheries. ICES Journal of Marine Science, 77: 1286–1294.

Redmon, J., and Farhadi, A. 2017. YOLO9000: better, faster, stronger. *In* Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7263–7271, Honolulu, HI.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. 2016. You only look once: unified, real-time object detection. *In* Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 779–788, Las Vegas, NV.

Ren, S., He, K., Girshick, R., and Sun, J. 2015. Faster R-CNN: towards real-time object detection with region proposal networks. *In* Proceedings on the Twenty-ninth Conference on Advances in Neural Information Processing Systems, pp. 91–99, Montreal, QC.

Richards, B. L., Beijbom, O., Campbell, M. D., Clarke, M. E., Cutter, G., Dawkins, M., Edgington, D. *et al.* 2019. Automated analysis of underwater imagery: accomplishments, products, and vision. A Report on the NOAA Fisheries Strategic Initiative on Automated Image Analysis 2014-2018. https://gitter.im/Russell91/TensorBox (last accessed 25 August 2020).

Salman, A., Siddiqui, S. A., Shafait, F., Mian, A., Shortis, M. R., Khurshid, K., Ulges, A. *et al.* 2020. Automatic fish detection in underwater videos by a deep neural network-based hybrid motion learning system. ICES Journal of Marine Science, 77: 1295–1307.

Shafait, F., Mian, A., Shortis, M., Ghanem, B., Culverhouse, P. F., Edgington, D., Cline, D. *et al.* 2016. Fish identification from

videos captured in uncontrolled underwater environments. ICES Journal of Marine Science, 73: 2737–2746.

Spampinato, C., Chen-Burger, Y.-H., Nadarajan, G., and Fisher, R. B. 2008. Detecting, tracking and counting fish in low quality unconstrained underwater videos. *In* Proceedings of the Third International Conference on Computer Vision Theory and Applications, Funchal, Madeira, Portugal.

Stehman, S. V. 1997. Selecting and interpreting measures of thematic classification accuracy. Remote Sensing of Environment, 62: 77–89.

Stewart, R. 2016. TensorBox: A Fast Object Detection Framework in TensorFlow. https://github.com/Russell91/TensorBox (last accessed 15 August 2019).

Stewart, R., Andriluka, M., and Ng, A. Y. 2016. End-to-end people detection in crowded scenes. *In* Proceedings of the Twenty-ninth IEEE Conference on Computer Vision and Pattern recognition, pp. 2325–2333, Las Vegas, NV.

Svellingen, C., Totland, B., White, D., and Øvredal, J. T. 2006. Automatic Species Recognition, Length Measurement and Weight Determination, Using the CatchMeter Computer Vision System. https://core.ac.uk/download/pdf/52043742.pdf (last accessed 25 August 2020).

Szegedy, C., Ioffe, S., and Vanhoucke, V. and Alemi, A. A. 2017. Inception-v4, inception-ResNet and the impact of residual connections on learning. *In* Proceedings of the Thirty-first AAAI Conference on Artificial Intelligence, San Francisco, CA.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. 2015. Going deeper with convolutions. *In* Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, Boston, MA.

Tseng, C., and Kuo, Y. 2020. Detecting and counting harvested fish and identifying fish types in electronic monitoring system videos using deep convolutional neural networks. ICES Journal of Marine Science, 77: 1367–1378.

Wallace, F., Williams, K., Towler, R., and McGauley, K. 2015. Innovative camera applications for electronic monitoring. *In* Fisheries Bycatch: Global Issues and Creative Solutions. Alaska Sea Grant, University of Alaska Fairbanks. http://doi. Org/10.4027/fbgics.2015.06 (last accessed 25 August 2020).

Wang, X., Shrivastava, A., and Gupta, A. 2017. A-Fast-RCNN: Hard positive generation via adversary for object detection. *In* Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2606–2615, Honolulu, HI.

Handling editor: Cigdem Beyan