

Applying Object Detection to Marine Data and Exploring Explainability of a Fully Convolutional Neural Network Using Principal Component Analysis

Herman Stavelin^a, Adil Rasheed^{a,b,*}, Omer San^c and Arne Johan Hestnes^d

^aNorwegian University of Science and Technology, Elektro D/B2, 235, Gløshaugen, O. S. Bragstads plass 2, Trondheim, Norway

^bMathematics and Cybernetics, SINTEF Digital, Kløbuveien 153, Trondheim, Norway

^cOklahoma State University, 201 General Academic Building Stillwater, Oklahoma 74078 USA

^dKongsberg Maritime, Strandpromenaden 50, Horten, Norway

ARTICLE INFO

Keywords:

Neural Networks
PCA
Object Detection
XAI
Machine Learning
YOLO


ABSTRACT

With the rise of focus on man made changes to our planet and wildlife therein, more and more emphasis is put on sustainable and responsible gathering of resources. In an effort to preserve maritime wildlife the Norwegian government decided to create an overview of the presence and abundance of various species of marine lives in the Norwegian fjords and oceans. The current work evaluates the possibility of utilizing machine learning methods in particular the You Only Look Once version 3 algorithm to detect fish in challenging conditions characterized by low light, undesirable algae growth and high noise. It was found that the algorithm trained on images collected during the day time under natural light could detect fish successfully in images collected during night under artificial lighting. The overall average precision score of 88% was achieved. Later principal component analysis was used to analyse the features learned in different layers of the network. It is concluded that for the purpose of object detection in specific application areas, the network can be considerably simplified since many of the feature detector turns out to be redundant.

1. Introduction

Coastal areas surrounding the Oslo fjord are some of the most populous areas in Norway which relies heavily on the well being of the fjord. Unfortunately the ecological condition is deteriorating leading to a decline in the population of several marine species. This has resulted in a need for greater environmental commitment to the area. The Norwegian government launched a project called Frisk Oslofjord - (Healthy Oslo Fjord) Frisk Oslofjord. Production of new knowledge and a basis for future management, test and verification of new technologies, and awareness creation are the major goals of the project. To this end one of the main activities in the project is to prepare detailed ecological maps of the Oslo fjord. These maps are expected to show the class of marine species and their locations at any particular time for better and accurate biomass estimation. Such information can be extremely useful for new upcoming technologies like Digital Twin Rasheed et al. (2020). As of today the mapping is conducted manually by inspecting images and then recording the findings. However, with the recent success of Artificial Intelligence (AI) and Machine Learning (ML) in image classification, text interpretation and big data analysis, new possibilities are opening up to automate the workflow. For example, Olsvik et al. (2019), Choi (2015), and Xu and Matzner (2018) have shown the power of computer vision and ML, not only in identifying, but also classifying

various marine species. Xu and Matzner (2018), utilizing You Only Look Once (YOLO) algorithm achieved a mean average precision score of 53.92% on a combination of three distinct datasets. Choi (2015) reported an F1-score of 84.8% for classification tasks. Detection of coral reef fishes in underwater images was studied using convolutional neural networks by Villon et al. (2016, 2018). In this work different post-processing decision rules were used to identify 20 fish species thereby taking into account the marine biodiversity in a cost-effective manner. The authors highlighted the promise of deep learning for monitoring fish biodiversity cheaply and effectively with an identification accuracy of 94.9%, which is greater than the rate of correct identification by humans (89.3%). Sung et al. (2017) reported 93% classification accuracy. Their study was based once again on the YOLO algorithm. Fish detection system under a variety of benthic background and illumination conditions was investigated by combining convolutional neural network (CNN) and long short-term memory (LSTM) networks by Labao and Naval Jr (2019). Combining principal component analysis (PCA), CNN and support vector machine (SVM) Sun et al. (2018) achieved a 95.18% fish recognition accuracy. More recently, Jalal et al. (2020), with an F1-score of 95.47 on the LIFECLEF 2015 (Joly et al. (2015)) claim to have outperformed all previous models used on the dataset. In another study Cai et al. (2020) proposes an approach combining YOLO version 3 (YOLOv3) with MobileNetV1 (Howard et al. (2017)) for fish detection in real breeding farm which can give accurate count of the fishes. These approaches, owing to the ease of automation, can allow mapping of the fjords and ocean in general, with much

 hermanstavelin@gmail.com (H. Stavelin); adil.rasheed@ntnu.no (A. Rasheed); osan@okstate.edu (O. San); arne.hestnes@km.kongsberg.com (A.J. Hestnes)

 www.adilrasheed.com (A. Rasheed)

ORCID(s):

higher spatio-temporal resolutions. However, despite the huge potential of exploitation of the ML based approach, the technology is not perfect Tu (1996) Abbe and Sandon (2018). Moniruzzaman et al. (2017) surveyed deep learning methods on underwater marine object detection and automated approaches for monitoring of underwater ecosystem. He concludes that the algorithms which can give super-human performance in image classification in good daylight might suffer to make correct classifications / detections in underwater scenarios where the visibility is highly diminished due to poor lighting conditions. Moreover there is little insight into the exact working of these algorithms something that is desirable to explain the detection capability of the algorithm. Thus there are two basic aspects that distinguish this work from the previous studies mentioned above:

- Evaluation of the performance of the model on noisy images recorded under poor lighting conditions with problematic algae growth on the camera lens. In the Norwegian fjord these are major issues.
- Explanation of the inner working of the algorithm. This is intended for optimizing the algorithm so that they can be deployed and trained quickly as and when required onboard.

We start this paper with a brief overview of the theory behind the algorithms used for object detection followed by information regarding the collection and processing of data. After this, results related to the prediction capability of the ML algorithm are presented followed by some insights into its inner workings. Lastly, the main take away from the current study is presented.

2. Theory

2.1. YOLO

The ML method utilized in this work is based on the YOLO architecture which is one of the most efficient and accurate algorithms for object detection in complicated scenes (Redmon and Farhadi (2018)). So far, the algorithm has been adopted for many applications including chemical sensing and detection of gas emission (Monroy et al. (2018)), anthracnose lesion detection on plant surfaces (Tian et al. (2019)), small target detection from drones (Xu et al. (2018)), traffic monitoring (Barthélemy et al. (2019)), plate recognition (Laroca et al. (2018)), pedestrian detection (Qu et al. (2018)), and autonomous driving (Choi et al. (2019)). Since, one of the objectives of the current study is to present an insight into the inner workings of the algorithm, we first give its brief but sufficient description. In a nutshell, YOLO is a Fully Convolutional Network (FCN) (Redmon and Farhadi (2018)). It uses a feature extractor with residual blocks consisting of 53 convolutional layers. A unique feature of this algorithm is that the detections are conducted at distinct layers throughout the network. In Figure 1 the entire structure of the network is shown. On the far left of the network one can see the layer through which the input images are fed in. This is followed by a gray box indicating YOLO's feature

extractors. The feature extractors, as the name implies, are responsible for bringing out important features from the input images. It consists of 23 residual blocks, each of which is built up of convolutional layers with 3×3 and 1×1 feature extractors. Batch normalization is applied in every convolutional layer to regularize the model, thus avoiding overfitting without the invocation of dropout (Redmon and Farhadi (2016)). The batch normalization is simply the normalization of the output of one layer by subtracting the batch mean and dividing by the batch standard deviation before passing it onto the next layer. 3×3 feature extractors with stride 2 are used when downsampling the feature maps. YOLO uses no form of pooling in contrast to most other FCNs (Zhao et al. (2018)). This is because pooling often results in a loss of low-level features (Kathuria (a)).

Since YOLO is a FCN, it is invariant to the size of the input images. However, for mere convenience (for example in batch processing of images and parallelization on GPUs) the dimensions of all the images are kept the same. Detections are made at layers 82, 94 and 106. By the time an input image transverses down to the first detection layer, its size shrinks by a factor of 32. Thus with an input image of size 416×416 the feature map at this layer will be 13×13 . After the first detection, the layer prior to the detection is upsampled by a factor of 2. In Figure 1, this corresponds to the last purple layer before the first orange layer. After a few more convolutional operations the resulting layer is concatenated with the feature map from an earlier layer having the identical size. In Figure 1, this corresponds to the concatenation of layers 61 and 86 to produce layer 87. The next detection is extracted at layer 94 after which the exact same procedure repeats once more. If the input image was 416×416 , the feature maps in layer 94 and later 106 would be of size 26 and 52, respectively. Extraction of detections at three locations within the network is an added feature of the third version of YOLO. According to the authors it improves the detection of small objects since it is able to capture more fine-grained features (Redmon and Farhadi (2018)). The output of the network is formulated as a three dimensional (3D) tensor and its dimensions are presented in Equation 1:

$$\text{Output} = S \times S \times [B * (5 + C)] \quad (1)$$

where S is the number of grid-cells, B the bounding boxes per grid cell and C the number of classes to detect. In Figure 2 an illustration of the feature map in a detection layer is presented. A bounding box is displayed as a red rectangle and the orange square is the grid cell that is at the center of the bounding box. This cell contains a long row of values. The pair (t_x, t_y) is the center of the box relative to the bounds of the grid cell the box belongs to. The pair (t_w, t_h) is the width and height of the box relative to the whole image. The confidence score p_o , sometimes called objectness score, tells us how certain it is that there is an object inside the box, and also how accurately the box encloses the object. Formally we have:

$$p_o = Pr(\text{Object}) * IOU_{pred}^{truth}. \quad (2)$$

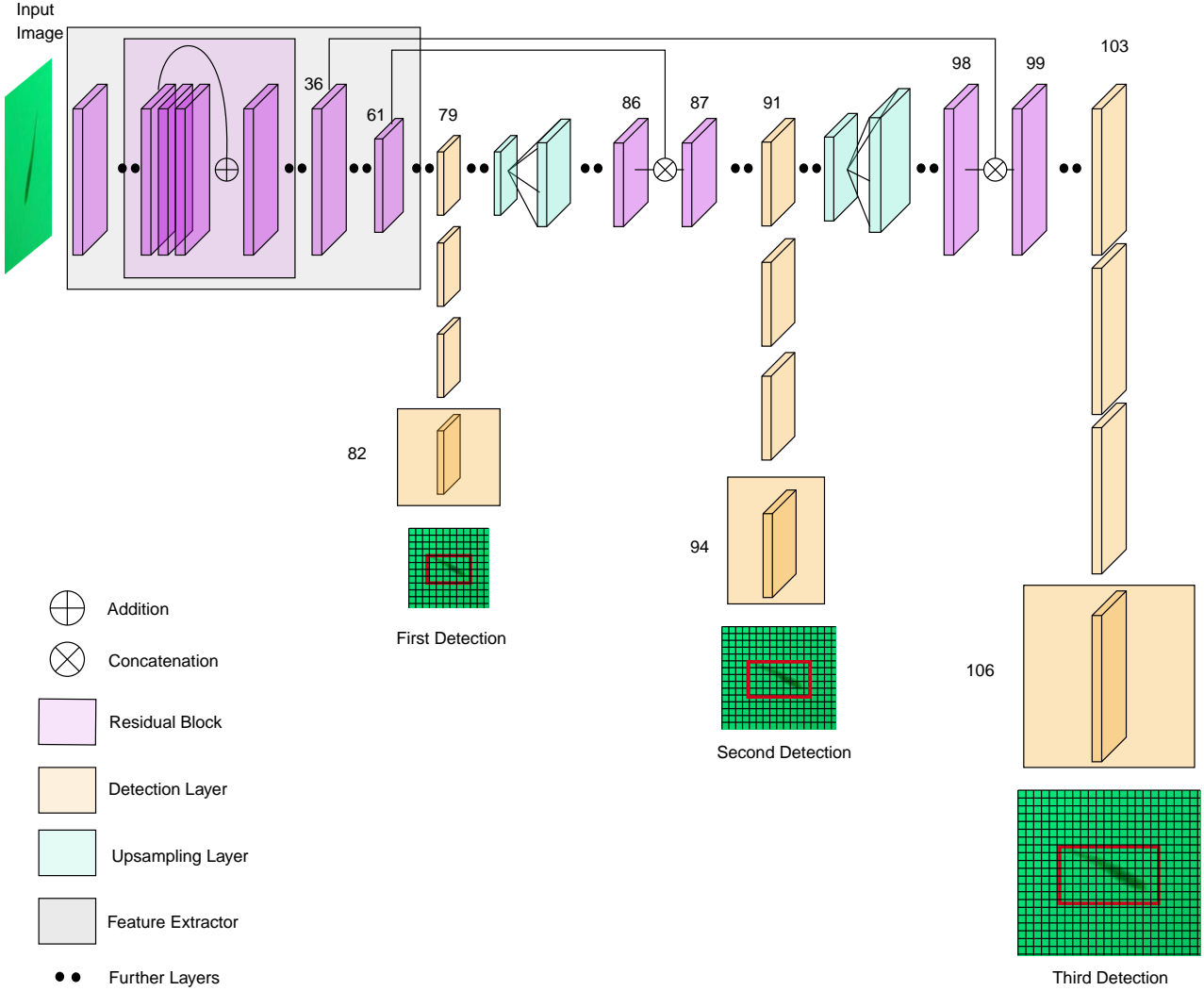


Figure 1: The structure of the entire YOLO v3 network. Illustration inspired by Kathuria (b)

where Intersection Over Union (IOU) is mathematically defined as:

$$\text{IOU} = \frac{\text{area of overlap}}{\text{area of union}} \quad (3)$$

IOU (Equation 3) is a measure of how much two shapes overlap. A high IOU means that the two shapes almost perfectly overlap each other. An IOU of zero would correspond to the two shapes not overlapping at all. If there is no object inside the box the IOU should be zero. If there is an object in the cell the confidence score should equal the IOU. The class probabilities p_i are formally defined as follows:

$$p_i = \text{Pr}(\text{Class}_i | \text{Object}) \quad (4)$$

This probability is called the Conditional Class Probability (CCP), and it is a measure of how likely it is, given that there is an object, which belongs to a certain class. Prior to the third version of YOLO, softmax activation was applied

on the output of the conditional class probabilities (Redmon et al. (2015)). This is now changed to the sigmoid activation function in the YOLO version 3.

2.2. Evaluation Metrics

There are many metrics that can be used to evaluate the performance of an ML algorithm. Here, we will present the most common, ubiquitous metrics. The definitions are obtained from Gopalakrishna et al. (2013), Flach (2019) and Everingham et al. (2010). In order to refresh the understanding of what the metrics convey in the current context, a simple scenario is constructed. Let us say that we have an algorithm that can predict the location and class of fish in images. The image in reality may or may not contain any fish but when it contains, the region containing the fish is also labelled. If a particular region contains a fish and the algorithm predicts it correctly, we have a so called True Positive (TP). If the algorithm does not detect any fish in the region

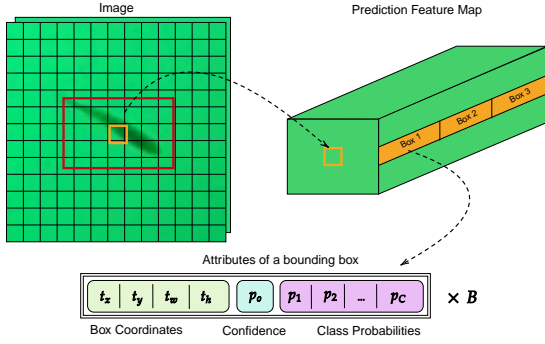


Figure 2: Explanation of YOLO's output tensor. Illustration inspired by Kathuria (a)

		Prediction	
		Fish	No Fish
Ground Truth	Fish	True positive	False negative
	No Fish	False positive	True negative

Figure 3: A confusion matrix relating TP, TN, FP and FN

and the labeled data confirms this then this is referred to as a True Negative (TN). False Positives (FP) tell that fish is detected by the algorithm when none existed in the labelled image. False Negatives (FN) tell that the algorithm failed to detect a fish that was actually there in the image. Figure 3 summarizes all these possibilities.

TP, TN, FP, FN are used in combination to construct other indicators like Precision, Recall, F1-score and Average Precision (AP) which are generally more concise and better measures of the performance of an algorithm. Precision, given by Equation 5, is a measure of how precise the predictions are. It gives us the percentage of predictions that agree with the ground truth as follows:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad (5)$$

Recall, given by Equation 6, tells us how good the algorithm is in finding all the TPs in an image, which is given as:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (6)$$

For most applications, one wants to find the parameter that leads to the best combined precision and recall. To this end, the F1-score (Equation 7) is designed as follows:

$$\text{F1-score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \quad (7)$$

Another important measure is the Average Precision (AP) score defined as

$$\text{AP@IOU} = \sum_n \frac{\text{Recall}_n - \text{Recall}_{n-1}}{\text{Precision}_n} \quad (8)$$

Here we note that there are many different formulations of AP. We have chosen to formulate it the same way that scikit-learn does (Pedregosa et al. (2011)). The @IOU refers to the fact that one must define when a prediction is accurate enough (i.e., TP). For example, how precisely a bounding box must encapsulate an object to be detected. This is done by setting an IOU threshold. Typically the AP is calculated for an $\text{IOU} \geq 0.5$. From Equation 8, it can be seen that calculating the AP is the same as calculating the area under the precision-recall curve.

2.3. Principal Component Analysis

As we have seen, the YOLO architecture consists of large number of hidden layers and within each layer there are multiple convolutional filters that are learned. Interpreting such a model boils down to analyzing the feature maps (which are the result of convolutional operation) extracted from each layer. A simple question that arises is that if so many filters and layers are actually required for the task of fish detection? To answer this question Principal Component Analysis (PCA) was applied on the feature maps extracted from each layer and analyzed. The PCA process provides an efficient way to compress data and explain variance of the data better than any other linear combination Jolliffe (2002) of the original feature maps. Although the geometric approach to PCA is due to Karl Pearson Pearson (1901), a more systematic approach to PCA is due to Harold Hotelling Hotelling (1933). In last few decades, there have been numerous studies to examine ways to nonlinearly generalize PCA. These methods often define a curve in latent space which minimize the mean squared error of all variables. Yet, the smoothness of the curve can be varied by the method. For example, an autoassociative or autoencoding neural network model Kramer (1991), Hsieh (2001, 2009) and a kernel PCA Schölkopf et al. (1998) can be considered two successful approaches of such a nonlinear PCA (NLPCA) framework. Moreover, other nonlinear dimensionality reduction techniques, such as principle curves Hastie and Stuetzle (1989), locally linear embedding Roweis and Saul (2000), isomap Tenenbaum et al. (2000) and self-organized map Kohonen (1982) approaches, can also be regarded as a discrete version of NLPCA. However, the NLPCA are themselves too complex to interpret and hence, within the current study, we have not considered these methods. PCA owing to its simplicity is chosen, and our findings as will be seen later justify the choice. Through this exercise there are basically two goals to be achieved:

1. To visualize as much of the information in the feature maps as possible, without displaying every single map.

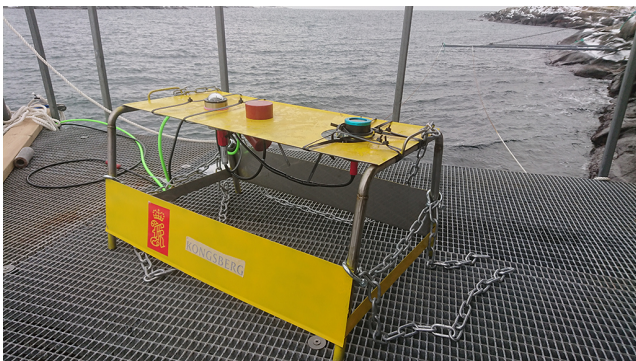


Figure 4: The measurement station. The leftmost glass dome is the camera. The red cylinder is the sonar and the blue and black cylinder on the right is the artificial lighting source

2. To find out if the feature maps contain truly distinct features, making them necessary for the network or the feature extractors used in the network are redundant.

The feature maps are naturally two dimensional (2D) so they are flattened and stacked as rows in a matrix which is subjected to PCA. If there are 32 feature maps of size 13×13 these are grouped together so as to have a matrix of dimensions $32 \times (13 * 13) = 32 \times 169$. PCA can then be applied to this matrix. The result of this is a list of components ordered based on the level of variance they capture. Two situations are worth noting: The first is when one component explains all the variance in the dataset. In this case the feature maps contain a clear pattern since they are all the same. The second extreme case is when all the components explain similar levels of variance. In this case there is no clear pattern in the feature maps, indicating that they are all very different from each other.

3. Data

The measurement station at Fulehuk in Norway can be seen in Figure 4. The station has a camera, a sonar and an artificial lighting source. It was deployed on the ocean floor about 30 meters below the water surface. It is oriented such that it looks from the ocean floor up at the water surface. The camera is a Goblin Shark and records 1080p at 30 fps with a horizontal angle of view of 92° in water Ivesdal. The sonar is a Simrad ES200-7CDK Split Kongsberg. The data was recorded between March and August 2019. The hardware was initially setup such that the camera and sonar would continuously capture data while artificial lighting would be enabled during nighttime. In order to lessen the data burden, images were uploaded to the storage container at 6 seconds intervals during March and much more infrequently (minutes to hours) during June, July and August. Furthermore, during March there was no artificial lighting during the night. For June, July and August artificial lighting was enabled both during the day and night time. A team of divers occasionally cleaned the camera lens to arrest excessive algae growth.

Parameters	Values
batch	64
subdivisions	16
width	416
height	416
channels	3
momentum	0.9
decay	0.0005
angle	0
saturation	1.5
exposure	1.5
hue	.1
learning rate	0.001
burn_in	1000
max_batches	500200
policy	steps
steps	400000, 450000
scales	.1,.1

Table 1
YOLO parameters used in this study

4. Implementation and Set-up

Setting up the YOLO network requires an initial set of weights and a configuration file. A pretrained set of weights called “darknet53.conv.74” was obtained from Redmon’s website Redmon. These weights were trained on the ImageNet dataset. The configuration file contains the entire network structure. All the general parameters are listed in Table 1. Width and height of the images were set to 416 to keep training time low. The number of filters in the three output layers had to be adjusted to accommodate the additional number of classes to be classified. Based on the article by Redmon and Farhadi (2018), the number of filters in the output layers should comply with the following equation:

$$\text{Filters}_n = (C + 5) * 3 \quad (9)$$

The YOLO algorithm was trained in two stages (see Figure 5). To prepare the labelled data for the first stage, 510 images from the month of March were selected and hand labelled to create a perfectly balanced dataset. LabelImg, as it supports the YOLO data format, was used to label the images Tzutalin (2019). None of the images corresponded to the night conditions under artificial lighting was included in the training set. The data set was split in the ratio 90:10 corresponding to the training and test sets. The trained model from the first stage was used to pseudo-label 3000 new images from the same month. While the correctly labelled images were used to augment the data as it is, the images corresponding to false positives and false negatives were manually corrected before including in the data set. Once again the additional data was divided into training and test sets in the same ratio as used earlier and the second stage training was conducted. Additional images from the months of June and July were also selected as test data to test the model performance beyond the month of March.

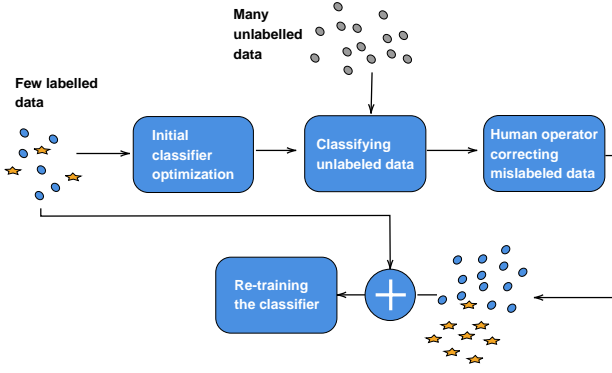


Figure 5: An overview of the training process

5. Results and Discussions

In Figure 6 a set of images containing fish from March to August can be seen. It is clear that the data quality varies a lot and that the size and illumination of the fish is not uniform. The fish consistently appears more distinct during the night because of the use of artificial lighting that gets reflected off their shiny surfaces. For the month of March the artificial lighting was still not operational resulting in pitch dark images recorded during the night. The growth of algae on the lens negatively impacted the image quality. The lens was cleaned from time to time but as can be seen in the images the cleaning was not fully effective. However, these are some of the practical limitations expected in the prevailing conditions in the Norwegian fjord. The norm for all YOLO-libraries is to display training per iterations and not per epochs. However, the conversion process from iterations to epochs is quite simple, and it is shown in Equation 10 as follows:

$$\text{epochs} = \frac{\text{batchsize} * \text{iterations}}{I_n} \quad (10)$$

The batch size is set to 64 and the number of images are given by I_n . Precision, recall and F1 are calculated at a lower confidence thresholds of 0.25.

Training Stage One: In Table 2, performance of the trained model from the first stage on the test data from the month of March is presented. Based on Equation 10, 1000 iterations correspond to roughly 140 epochs. Beyond the 2000 iterations the gain in accuracy was only marginal. However, the training was continued for 4000 iterations to ensure that no substantial improvements could be made beyond it. It was observed that the metrics started fluctuating beyond it. The best metrics were obtained at the iteration number 4200. At this point the trained model had a recall of 69%, precision of 74% and F1 score of 72%.

Training Stage Two: In Table 3 the progress during the training at stage two, where ~ 2700 images were used is shown. Due to the increased dataset size, 1000 iterations corresponded to roughly 24 epochs. It can be seen from the

Iterations	AP@50	Precision	Recall	F1
1000	0.3881	0.53	0.50	0.51
2000	0.6003	0.72	0.64	0.68
3000	0.5974	0.72	0.61	0.66
4000	0.6015	0.73	0.62	0.67
4200	0.6338	0.74	0.69	0.72

Table 2

Performance of the trained model from stage two on the test data consisting of 51 images

Iterations	AP@50	Precision	Recall	F1
1000	0.8118	0.83	0.81	0.82
2000	0.8274	0.79	0.84	0.81
3000	0.8595	0.83	0.88	0.85
4000	0.8679	0.82	0.87	0.85
5000	0.8475	0.85	0.84	0.84
-				
8000	0.8809	0.83	0.87	0.85

Table 3

Performance of the trained model from stage two on the test data consisting of 300 images

Table 3 that the increase in the size of the dataset greatly improves all the performance metrics, achieving a AP of 0.88 and F1-score of 0.85 despite the noisy dataset. It is, however, reasonable to think that better results could be achieved if the the training data quality could be improved. The limiting quality of the data is probably due to the authors inability to correctly label the data by hand. This is because of the fact that in some images it was almost impossible to distinguish fish from other objects leading to ambiguities in around 5% of the hand labelled images. Unfortunately there was no better way to improve the labelling. We can only assume that our ability to correctly classify fishes with 95% surpassed human accuracy of 89.3% reported by Villon et al. (2016, 2018). The network at stage two was initialized with the trained weights from first stage. It is observed that most of the learning once again happens in the first 1000 iterations. The training was run for another 7000 iterations but without any significant improvement in the performance. At this stage the precision, recall and F1 score was 83%, 87% and 85% respectively.

5.1. Visual Predictions on Unseen Images

Prediction on the data from March: In Figure 7 some detections made on unseen data from March can be seen. It is clear that the algorithm is able to satisfactorily detect the fishes in the image. This is to be expected as an F1-score of 0.85 was achieved after the second round of training. When inspecting all the data from March it was noticed that the images throughout the month were similar and hence the training set was indeed representative of the test set. This is reflected in the performance of the algorithm on unseen data from March as seen in Figure 7.

Prediction on data from June and July: When the model trained was applied to data from other months (eg. June and

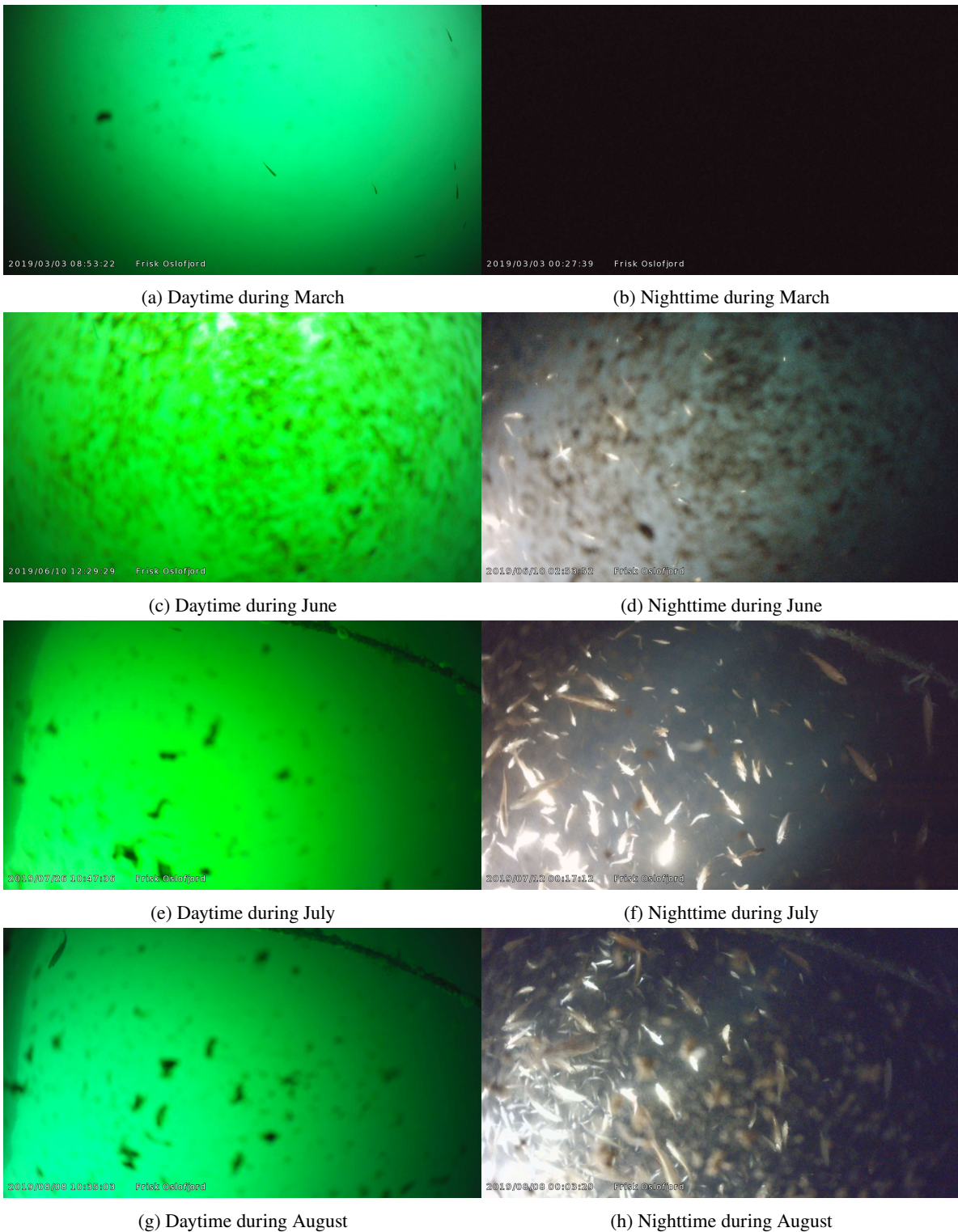


Figure 6: Some samples from the different time periods. We have good conditions during March, but complete darkness at night. During June there is large amounts of algae on the lens. Between June and July the lens was cleaned and the camera angle changed

July), mixed results were obtained as can be seen in Figure 8. On the left hand side of Figures 8a, 8b and 8c it looks like twigs or perhaps pieces of plastic from the rig is being marked as fishes. Another example of this is in Figure 8a

where the eye of a large fish is marked as a fish. This indicates that the trained model is not sufficiently robust to outliers, noise and significantly changed conditions during these months compared to March. One positive observation

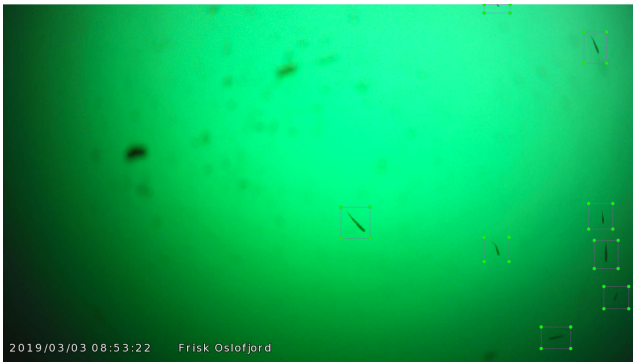


Figure 7: Image from 03.03.2019 with predicted bounding boxes

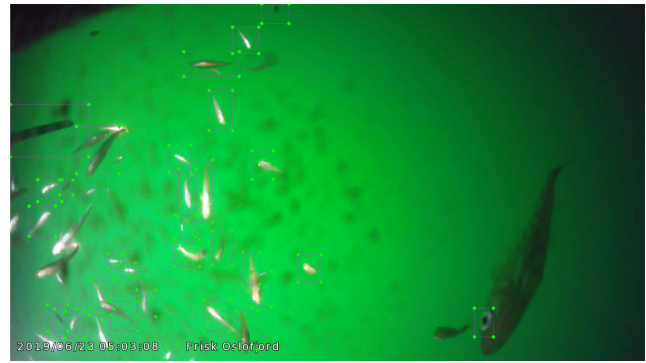
is that in many of the images (eg. Figure 8), a large number of fish got successfully detected both during the day and during the night. Especially in Figure 8b one can notice that the trained model did fairly well in detecting every tiny fish in abundance. However, at the same time there are also a reasonable number that went undetected.

In Figure 9 some examples where the algorithm performed particularly poorly are presented. One can see that sometimes even large fishes can go undetected. Another problem that is encountered is that occasionally multiple bounding boxes are allocated to the same fish (Figure 9b). Furthermore, the rope seen in this image is being detected as a fish. From this we know that the algorithm does not have to see an entire fish to mark it as a fish. This is good in the sense that it can detect fish that are not completely within the field of view of the camera but leads to several errors as shown here. The algorithm rarely marks algae as fish which is impressive. For example in Figure 9b it could have been a possibility to mistake algae for fish. Why the algorithm does not do this is unclear. Perhaps it uses the blurriness of the algae to determine they are not fish or perhaps it could be attributed to the overall shape of the algae. Last but not the least it should be noted that the algorithm is able to detect fish during a variety of lighting conditions as seen in Figure 8. As explained in Section 4, the algorithm had never seen images under artificial lighting conditions during the training. In most of the images the bounding boxes are correctly placed around the fishes. We remark that the precision is very high while recall is quite low. This could be attributed to the following two reasons:

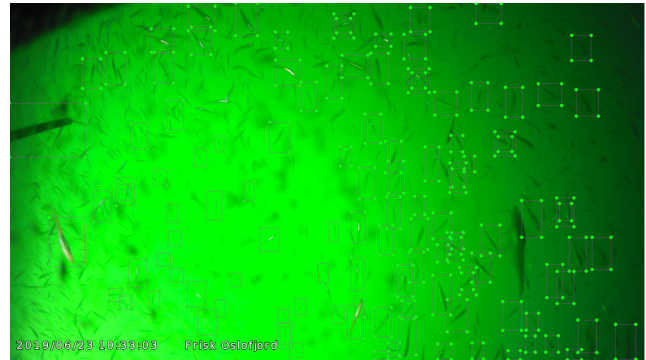
1. The images in the test set are quite different from those in the training data. The different environmental conditions unseen during the training step disturb the predictive ability of the model.
2. When images are input to YOLO they are downscaled to 416×416 . This results in a loss of resolution and hence a loss of information required for correct detection and classification.

5.2. Insights into the Inner Workings of the FCN

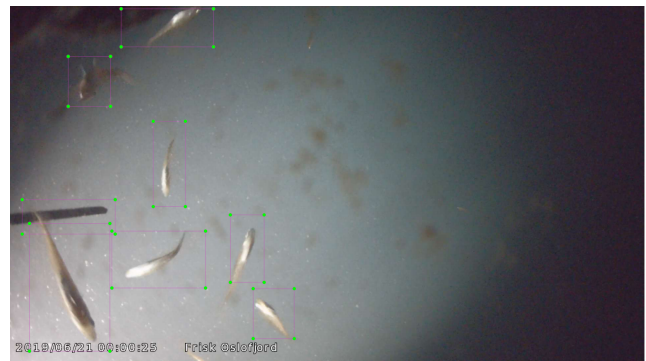
In order to understand the inner workings of the network, feature maps from the hidden layers were extracted.



(a)



(b)



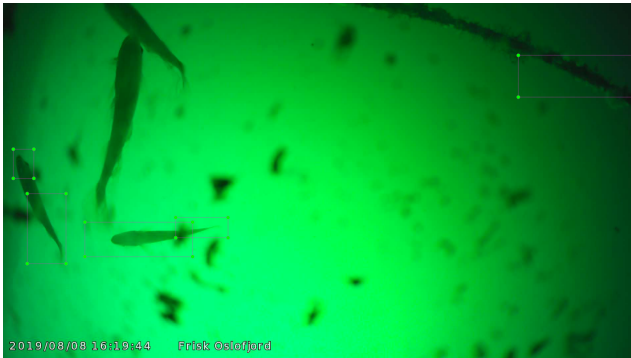
(c)

Figure 8: Some predictions on data sampled from 23. June 2019

From the description of the feature extractor in Redmon and Farhadi (2018) it is known that the first layer of the network has 32 filters of size 3×3 . In reality these filters are $3 \times 3 \times 3$ because color images have three channels. It should be noted that 32 is actually the lowest number of filters in any layer. Some layers in the network has up to 1024 filters making the task of visualizing and interpreting them individually almost impossible. It is worth stressing that the result produced by the convolution operation on the colored images do not actually produce feature maps that can be visualized in a comprehensible way. This is because the resulting matrix values are not confined to the 0-255 interval. Therefore, to actually create visualisations, the values were normalized and the default colormap “viridis” from Matplotlib was applied. The colormap maps low values to dark blue and high values to



(a) Nighttime 10.07.2019



(b) Daytime 08.08.2019

Figure 9: Some especially poor detections from July and August

yellow. The image that was fed as input to the trained network from which intermediate feature maps were extracted is given by Figure 7. In Figure 10 we see some plots of the intermediate feature maps produced in the very first convolutional layer of the network. Based on these maps it seems that the filters produce every imaginable variant of the feature map. Some filters blur the images while others sharpen it. Some even seems to produce the negative. In Figure 10d there are strong gradients highlighting edges while in Figure 10e the image is almost completely smooth. In Figures 10c and 10d one can see that edges are detected on opposite sides. In Figure 10c left-edges are detected, while in Figure 10d right-edges are detected. In Figures 10a and 10b we see inverse values. Using the negative might be one of the reasons why the network seems to detect fish both with and without the presence of artificial lighting. This also explains why it doesn't matter to the network whether the fishes are dark or light in color.

In Figure 11 images reconstructed in deeper layers are displayed. When comparing Figures 10 and 11 one notices that the images have different aspect ratios. When looking at the results from the first layer one can look at the results using the original aspect ratio of the images that were put in. However, as an image passes through the network, more and more information seems to “bleed” onto the originally unused top and bottom margins of the image. It seems like YOLO tries to store as much information as possible.

In the layers closer to the input layer (e.g. Figures 11a,

11b, 11c) one can see that the images closely resemble the original image. Gradually as one traverses through the layers the images become coarser. In Figures 11e and 11f one can see that the fish and some algae are very prominent. In these images the algae is brighter than the fish. It might be that high values in these specific images in these specific layers means that an object should be ignored, as labels around algae are undesirable. In Figure 11k a reconstructed image from the 81st layer of the network is shown. This layer consists of 1024 feature maps of size 13×13 . It is based on the images in this layer that the first prediction is made. When one has traversed this far into the network it is virtually impossible to recognize what information the different pixels encode. However, it is to be noted that the original aspect ratio border seems to have vanished, and that information seems to be stored in the entire image. One can recall that YOLO makes detections at three different scales. As one moves past the first detection layer the network starts to scale up the image. In Figure 11l one can see an image from the layer before the second detection. It is still virtually impossible for a human to harness any meaningful information from these images. However, one sees that some horizontal lines have started to appear. It seems like YOLO is starting to reconstruct the original image. After the second detection layer YOLO further scales up the image. In the four layers prior to the third and last detection, the images start to make a bit more sense. In Figure 11m one can recognize blobs that correspond to the fish that is to be detected. It is evident that YOLO is able to “remember” what the original image looked like. From this layer onwards, and to the end, these blobs become more and more distinct as can be seen in Figures 11n, 11o and 11p. Thus for the very last detection layer the network can perhaps create bounding boxes around the brightest pixels. Furthermore, one can notice that in Figure 11p the fishes are represented by blocks of bright and dark pixels together. It can also be observed that the restored top and bottom margins of these images contain very little variance. It seems that in this detection layer most of the information is retrieved from the values within the original aspect ratio.

The discussion so far is based on a handful of images extracted from each layer. As explained earlier there can be up to 1024 feature maps in some of the layers which are humanly impossible to interpret. In order to develop some statistical understanding of different layers we conducted PCA on the feature maps extracted from each layer individually Mishra et al. (2017). Figure 12 gives the plots of the ratio of variance for the 5 most prominent principal components. It appears that for our early layers, as can be seen in Figures 12a and 12b, almost all the variance can be explained using a single component. Gradually, as one moves through the layers it seems that more and more information is spread out across the feature maps within a layer. In other words the feature maps become more and more distinct within each layer. All the feature maps in the first layer are quite alike, while the feature maps are all very different in the 81st layer. However, in the next layer (Figure 12f), a mode collapse is

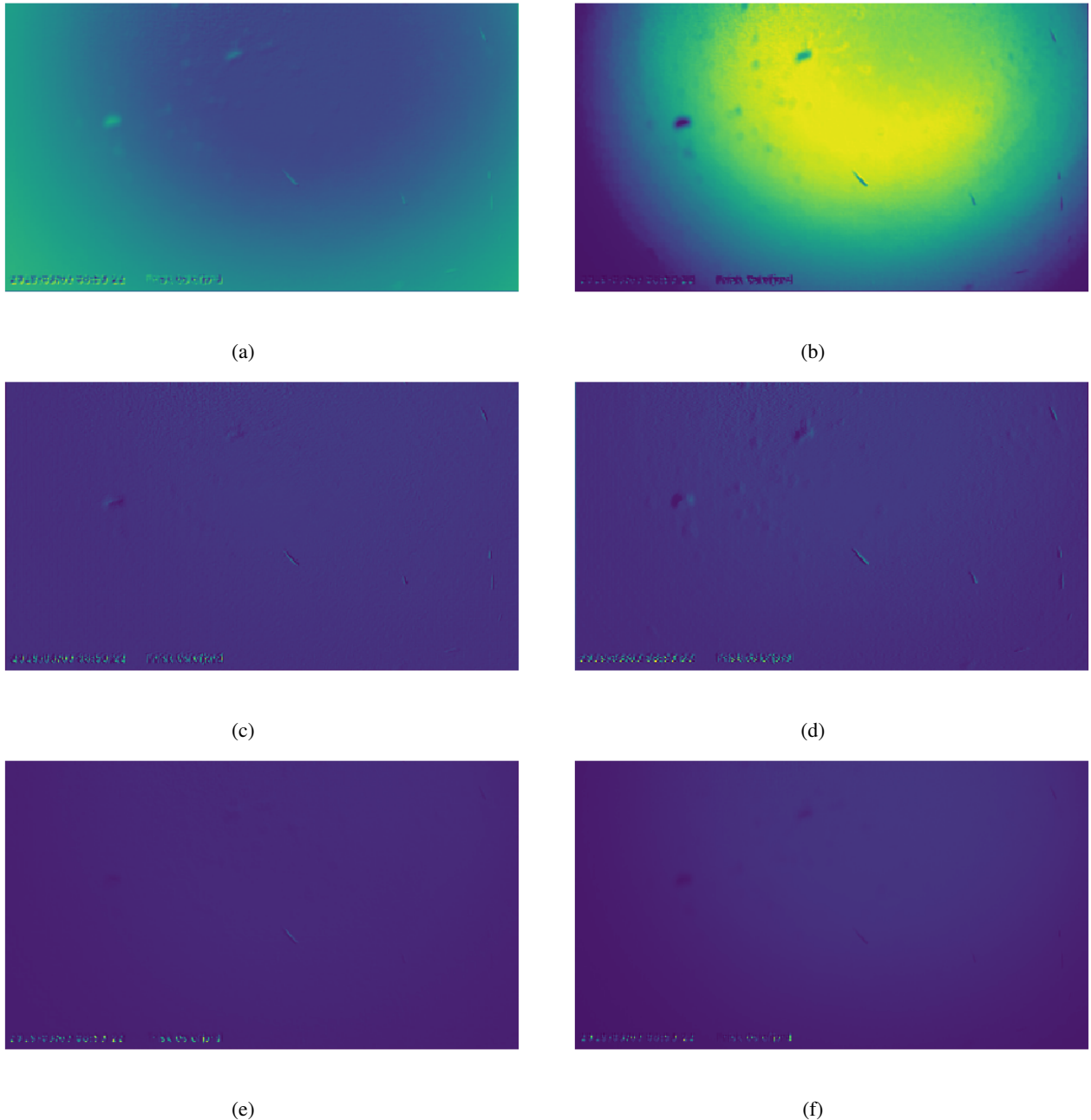


Figure 10: Images extracted from the first convolutional layer in the network

observed. Seemingly, the output filters are able to extract some pattern from the 81st layer. If all the components were equally contributing that could indicate that no pattern was found in the data. Perhaps the filters in the early layers perform similar operations, while in the later layers the operations become more specialised. If this is the case it would make sense that the deeper layers contain more distinct data. We recall that YOLO is a large FCN capable of detecting and classifying several thousand classes simultaneously and that it might be an overkill to just detect one class, as is being done here. Perhaps PCA would give very different results if the network was trained on a different dataset with more classes. It might be that this kind of PCA could be used

as an optimization technique on FCNs. If most of the variance is explained by one, or a few, principal components, perhaps the number of filters in that layer could be reduced. This could be implemented to reduce run times of FCNs. This could be done by first training the network and then running PCA and reducing the amount of filters in the layers that are mostly explained by a few principal components. Then the network could be retrained and PCA re-calculated. This could be done until the accuracy starts to drop below a certain threshold, in relation to the original accuracy of the network. This would greatly increase training times, but could speed up test times while still maintaining almost the same accuracy as the original network. This could in addi-

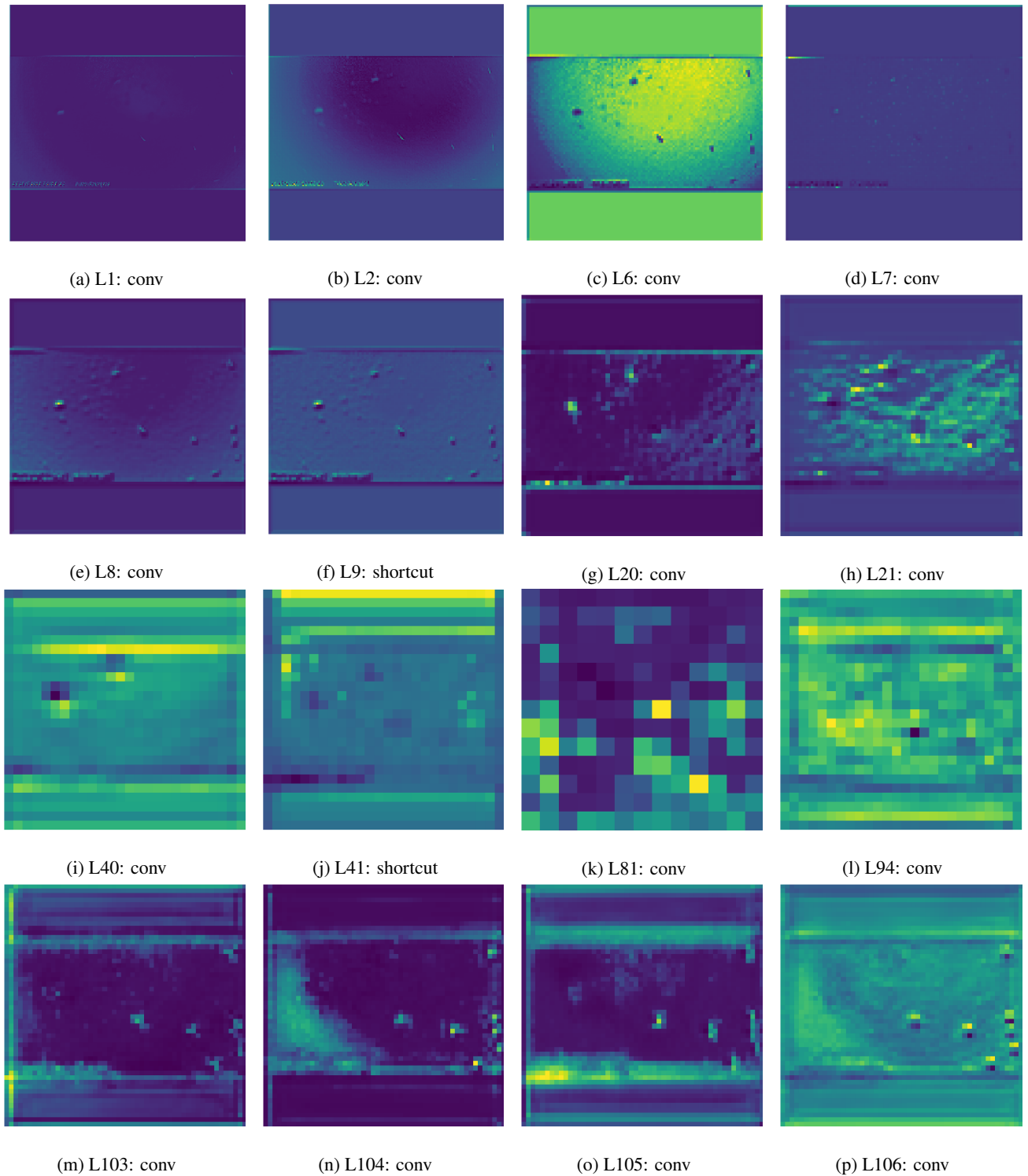


Figure 11: Feature maps from intermediate layers in YOLO. Detections are made at layer 82, 94 and 106

tion make it easier to interpret and explain the network as a simplified network is nevertheless easier to analyze.

In Figure 13 one can see visualizations made by constructing images from only the most, and second most important components produced by the PCA. In Figure 12a we observe that by just using the first component we retain almost all the variance. This is in correspondence with what

we see in Figures 13a and 13b. There is seemingly very little information in the second image that one can not be found in the first. We see from Figures 13c and 13d that the images have started to become more distinct. In Figure 12c we see that about 50% of the variance in this image is explained by the first principal component. In the 81st layer, as seen in Figures 13e and 13f, we see that the principal components

Object Detection on Marine Data

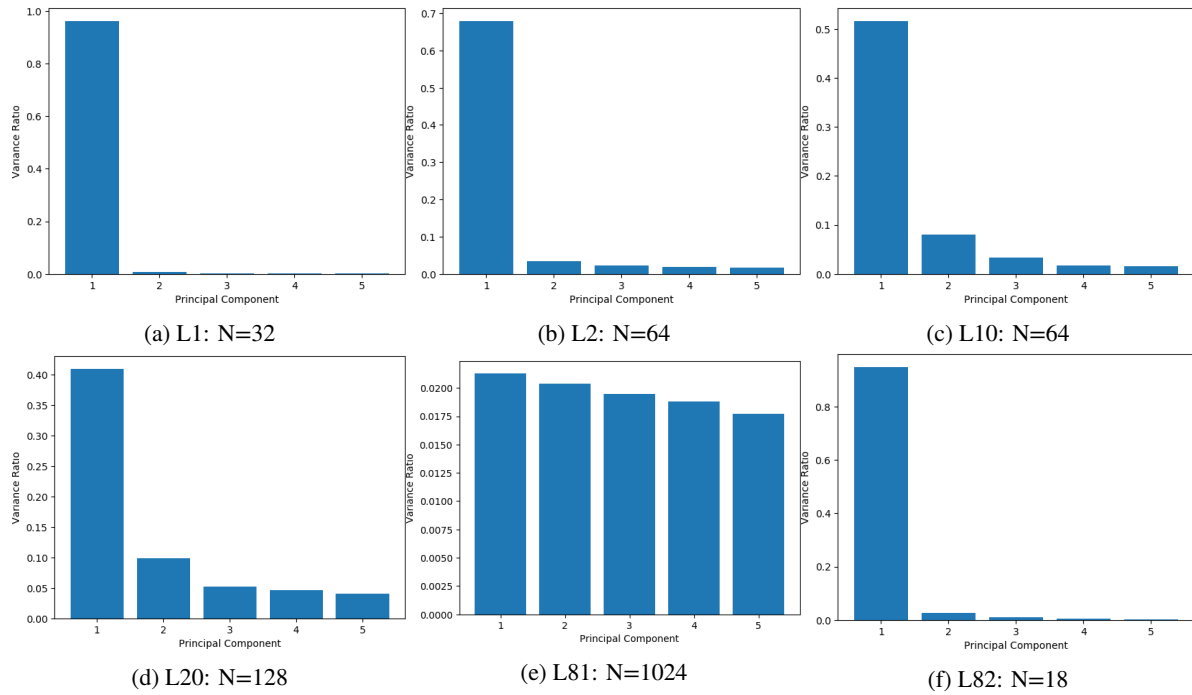


Figure 12: Bar plots: PCA analysis of intermediate layers. N is the number of feature maps.

starts to hone in on certain regions. This is especially interesting when we look at a normal image from the same layer, as seen in Figure 11k, which mostly looks like noise. In this layer the principal components explain almost the same amount of variance as can be seen in Figure 12f.

6. Conclusion

In this paper we utilized YOLO to detect fish in images recorded under water and provided insight into the internal workings of the algorithm. The major findings of the project can be enumerated as follows:

- The work presented in this report makes a significant contribution to the Healthy Oslo fjord project by building a workflow that can be used to generate labelled data from images using semi-supervised learning. The current trained model already has a F1-score of 0.85 and has been used to generate labelled dataset that will also provide labels to the sonar data that was acquired during the same field campaign.
- We showed that using YOLO as a semi-supervised algorithm bears some merit. Using a tiny amount of hand labeled data to pseudo-label a lot of data greatly reduced the manual labor involved.
- One important conclusion is that the trained algorithm was robust against the lighting conditions. The model trained on images taken during the day could still perform well on images taken under artificial lighting conditions at night.

- Internal layers of YOLO network were illustrated and PCA was utilized to extract information from the thousands of filters used in the network. Investigating the variance ratio vs principle component plots for the different layers, we observed that most of the variance in the reconstruction of the images using the trained filters in the first few layers can be explained by a single component. The number of components required to explain the variance in the deep layers gradually increases. This hints at the fact that for detecting single class object (fish in this case)

Despite the interesting results presented in the article, there are several aspects in the current work that requires more in-depth investigation. We used a very complicated network to detect only a single class (of fish). It would be interesting to see how the network behaves if it is utilized to do multiclass object detection and classification. Also it will be valuable to explore the possibility of employing PCA to optimize the network.

Acknowledgement

The authors would like to thank the Healthy Oslo Fjord initiative for providing the funds that enabled creating the dataset which we have presented here.

References

- Abbe, E., Sandon, C., 2018. Provable limitations of deep learning. [arXiv:1812.06369](https://arxiv.org/abs/1812.06369).
- Barthélemy, J., Verstaavel, N., Forehead, H., Perez, P., 2019. Edge-computing video analytics for real-time traffic monitoring in a smart city. *Sensors* 19, 2048.

Object Detection on Marine Data

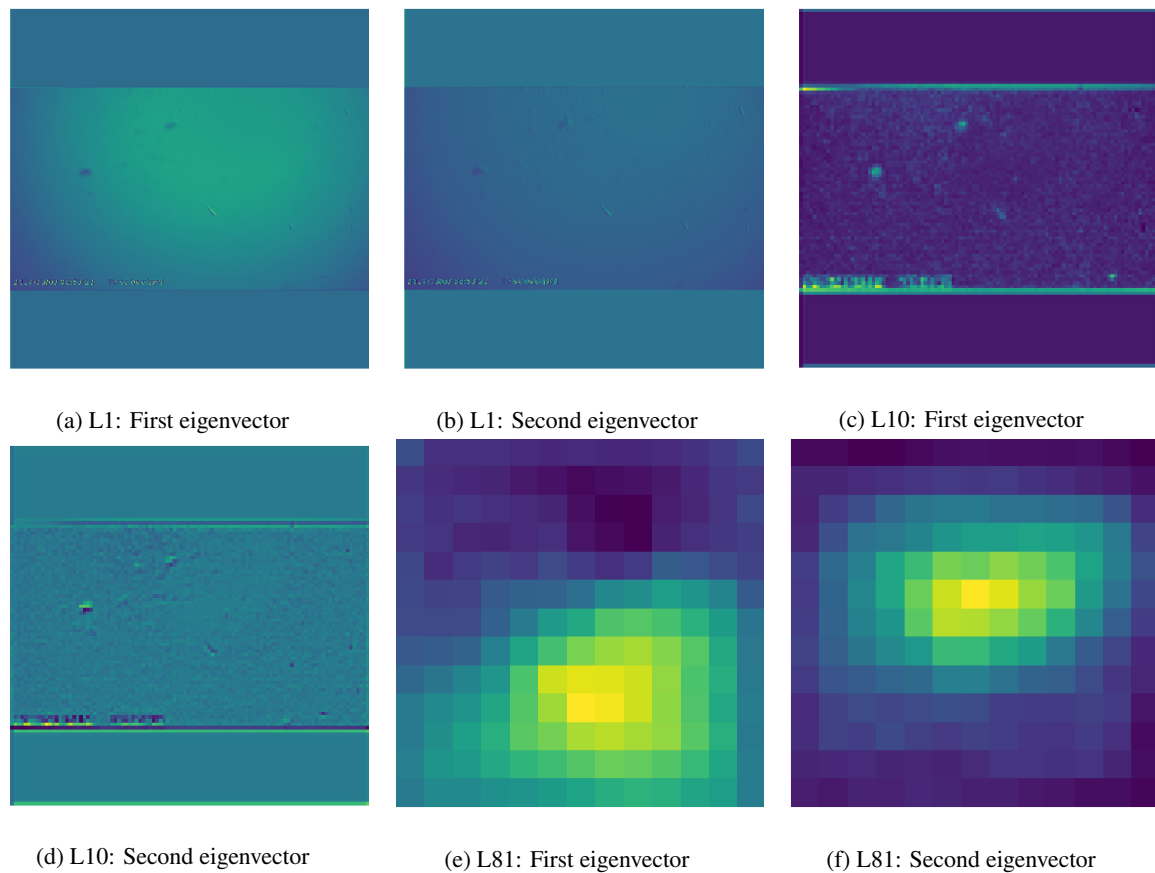


Figure 13: Images: PCA analysis of intermediate layers

- Cai, K., Miao, X., Wang, W., Pang, H., Liu, Y., Song, J., 2020. A modified yolov3 model for fish detection based on mobilenetv1 as backbone. *Aquacultural Engineering* 91, 102117. URL: <http://www.sciencedirect.com/science/article/pii/S0144860920301631>, doi:<https://doi.org/10.1016/j.aquaeng.2020.102117>.
- Choi, J., Chun, D., Kim, H., Lee, H.J., 2019. Gaussian yolov3: An accurate and fast object detector using localization uncertainty for autonomous driving, in: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 502–511.
- Choi, S., 2015. Fish identification in underwater video with deep convolutional neural network: SNUMedinfo at LifeCLEF fish task 2015.
- Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A., 2010. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision* 88, 303–338.
- Flach, P., 2019. Performance evaluation in machine learning: the good, the bad, the ugly and the way forward. <https://aaai.org/ojs/index.php/AAAI/article/view/5055>.
- Frisk Oslofjord, . Om frisk oslofjord. <https://friskoslofjord.no/om-frisk-oslofjord/>.
- Gopalakrishna, A.K., Ozcelebi, T., Liotta, A., Lukkien, J.J., 2013. Relevance as a metric for evaluating machine learning algorithms, in: Perner, P. (Ed.), *Machine Learning and Data Mining in Pattern Recognition*, Springer Berlin Heidelberg, Berlin, Heidelberg. pp. 195–208.
- Hastie, T., Stuetzle, W., 1989. Principal curves. *Journal of the American Statistical Association* 84, 502–516.
- Hottelling, H., 1933. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology* 24, 417.
- Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H., 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications. [arXiv:1704.04861](https://arxiv.org/abs/1704.04861).
- Hsieh, W.W., 2001. Nonlinear principal component analysis by neural networks. *Tellus A* 53, 599–615.
- Hsieh, W.W., 2009. *Machine learning methods in the environmental sciences: Neural networks and kernels*. Cambridge University Press, New York.
- Ivesdal, I., . Imenco: Goblin Shark. <https://imenco.no/product/imenco-goblin-shark-ip-hd-wide-angle-overview-subsea-video-camera/>.
- Jalal, A., Salman, A., Mian, A., Shortis, M., Shafait, F., 2020. Fish detection and species classification in underwater environments using deep learning with temporal information. *Ecological Informatics* 57, 101088. URL: <http://www.sciencedirect.com/science/article/pii/S1574954120300388>, doi:<https://doi.org/10.1016/j.ecoinf.2020.101088>.
- Jolliffe, I.T., 2002. *Principal components analysis*. Springer, Berlin.
- Joly, A., Goëau, H., Glotin, H., Spampinato, C., Bonnet, P., Vellinga, W.P., Planqué, R., Rauber, A., Palazzo, S., Fisher, B., Müller, H., 2015. Lifeclef 2015: Multimedia life species identification challenges, in: *Proceedings of the 6th International Conference on Experimental IR Meets Multilinguality, Multimodality, and Interaction - Volume 9283*, Springer-Verlag, Berlin, Heidelberg. p. 462–483. URL: https://doi.org/10.1007/978-3-319-24027-5_46, doi:[10.1007/978-3-319-24027-5_46](https://doi.org/10.1007/978-3-319-24027-5_46).
- Kathuria, A., a. How to implement a YOLO object detector in PyTorch. <https://blog.paperspace.com/how-to-implement-a-yolo-object-detector-in-pytorch/>.
- Kathuria, A., b. YOLOv3 Object Detection. <https://towardsdatascience.com/yolo-v3-object-detection-53fb7d3bfe6b>.
- Kohonen, T., 1982. Self-organized formation of topologically correct feature maps. *Biological Cybernetics* 43, 59–69.
- Kongsberg, . [https://www.simrad.com/www/01/NOKBG0397.nsf/AllWeb/88E4199935B57F65C12581E70031DFDF/\\$file/429713aa_es200_7cdk_split_data_sheet_english.pdf?OpenElement](https://www.simrad.com/www/01/NOKBG0397.nsf/AllWeb/88E4199935B57F65C12581E70031DFDF/$file/429713aa_es200_7cdk_split_data_sheet_english.pdf?OpenElement).
- Kramer, M.A., 1991. Nonlinear principal component analysis using autoassociative neural networks. *AICHe Journal* 37, 233–243.
- Labao, A.B., Naval Jr, P.C., 2019. Cascaded deep network systems with linked ensemble components for underwater fish detection in the wild.

- Ecological Informatics 52, 103–121.
- Laroca, R., Severo, E., Zanlorensi, L.A., Oliveira, L.S., Gonçalves, G.R., Schwartz, W.R., Menotti, D., 2018. A robust real-time automatic license plate recognition based on the YOLO detector, in: 2018 International Joint Conference on Neural Networks (IJCNN), IEEE. pp. 1–10.
- Mishra, S.P., Sarkar, U., Taraphder, S., Datta, S., Swain, D.P., Saikhom, R., Laishram, M., 2017. Multivariate statistical data analysis-principal component analysis (PCA). *International Journal of Livestock Research* 7, 60–78.
- Moniruzzaman, M., Islam, S.M.S., Bennamoun, M., Lavery, P., 2017. Deep learning on underwater marine object detection: a survey, in: International Conference on Advanced Concepts for Intelligent Vision Systems, Springer. pp. 150–160.
- Monroy, J., Ruiz-Sarmiento, J.R., Moreno, F.A., Melendez-Fernandez, F., Galindo, C., Gonzalez-Jimenez, J., 2018. A semantic-based gas source localization with a mobile robot combining vision and chemical sensing. *Sensors* 18, 4174.
- Olsvik, E., Trinh, C.M.D., Knausgård, K.M., Wiklund, A., Sjørdalen, T.K., Kleiven, A.R., Jiao, L., Goodwin, M., 2019. Biometric fish classification of temperate species using convolutional neural network with squeeze-and-excitation. *arXiv:1904.02768*.
- Pearson, K., 1901. Principal components analysis. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 6, 559.
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E., 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research* 12, 2825–2830.
- Qu, H., Yuan, T., Sheng, Z., Zhang, Y., 2018. A pedestrian detection method based on YOLOv3 model and image enhanced by Retinex, in: 2018 11th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI), IEEE. pp. 1–5.
- Rasheed, A., San, O., Kvamsdal, T., 2020. Digital twin: Values, challenges and enablers from a modeling perspective. *IEEE Access* 8, 21980–22012.
- Redmon, J., Divvala, S., Girshick, R., Farhadi, A., 2015. You only look once: Unified, real-time object detection. *arXiv:1506.02640*.
- Redmon, J., Farhadi, A., 2016. Yolo9000: Better, faster, stronger. *arXiv:1612.08242*.
- Redmon, J., Farhadi, A., 2018. Yolov3: An incremental improvement. *arXiv:1804.02767*.
- Redmon, J.C., . YOLO website. URL: <https://pjreddie.com/>.
- Roweis, S.T., Saul, L.K., 2000. Nonlinear dimensionality reduction by locally linear embedding. *Science* 290, 2323–2326.
- Schölkopf, B., Smola, A., Müller, K.R., 1998. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation* 10, 1299–1319.
- Sun, X., Yang, J., Wang, C., Dong, J., Wang, X., 2018. Low-contrast underwater living fish recognition using pcanet, in: Ninth International Conference on Graphic and Image Processing (ICGIP 2017), International Society for Optics and Photonics. p. 106150Y.
- Sung, M., Yu, S., Girdhar, Y., 2017. Vision based real-time fish detection using convolutional neural network, in: OCEANS 2017 - Aberdeen, pp. 1–6. doi:10.1109/OCEANSE.2017.8084889.
- Tenenbaum, J.B., De Silva, V., Langford, J.C., 2000. A global geometric framework for nonlinear dimensionality reduction. *Science* 290, 2319–2323.
- Tian, Y., Yang, G., Wang, Z., Li, E., Liang, Z., 2019. Detection of apple lesions in orchards based on deep learning methods of cyclegan and yolov3-dense. *Journal of Sensors* 2019.
- Tu, J.V., 1996. Advantages and disadvantages of using artificial neural networks versus logistic regression for predicting medical outcomes. *Journal of Clinical Epidemiology* 49, 1225 – 1231.
- Tzutalin, 2019. Labelimg. <https://github.com/tzutalin/labelImg>.
- Villon, S., Chaumont, M., Subsol, G., Villéger, S., Claverie, T., Mouillot, D., 2016. Coral reef fish detection and recognition in underwater videos by supervised machine learning: Comparison between Deep Learning and HOG+ SVM methods, in: International Conference on Advanced Concepts for Intelligent Vision Systems, Springer. pp. 160–171.
- Villon, S., Mouillot, D., Chaumont, M., Darling, E.S., Subsol, G., Claverie, T., Villéger, S., 2018. A deep learning method for accurate and fast identification of coral reef fishes in underwater images. *Ecological Informatics* 48, 238–244.
- Xu, W., Matzner, S., 2018. Underwater fish detection using deep learning for water power applications, in: 2018 International Conference on Computational Science and Computational Intelligence (CSCI), IEEE. pp. 313–318.
- Xu, Z., Shi, H., Li, N., Xiang, C., Zhou, H., 2018. Vehicle Detection Under UAV Based on Optimal Dense YOLO Method, in: 2018 5th International Conference on Systems and Informatics (ICSAI), IEEE. pp. 407–411.
- Zhao, Z.Q., Zheng, P., tao Xu, S., Wu, X., 2018. Object detection with deep learning: A review. *arXiv:1807.05511*.



Herman Stavelin is a Masters student in Robotics and Cybernetics at the Norwegian University of Science and Technology. He is currently writing a Masters Thesis on Object Detection and Explainable AI.



Omer San received his bachelors in aeronautical engineering from Istanbul Technical University in 2005, his masters in aerospace engineering from Old Dominion University in 2007, and his Ph.D. in engineering mechanics from Virginia Tech in 2012. He worked as a postdoc at Virginia Tech from 2012-'14, and then from 2014-'15 at the University of Notre Dame, Indiana. He has been an assistant professor of mechanical and aerospace engineering at Oklahoma State University, Stillwater, OK, USA, since 2015. He is a recipient of U.S. Department of Energy 2018 Early Career Research Program Award in Applied Mathematics. His field of study is centered upon the development, analysis and application of advanced computational methods in science and engineering with a particular emphasis on fluid dynamics across a variety of spatial and temporal scales.



Adil Rasheed is the professor of Big Data Cybernetics in the Department of Engineering Cybernetics at the Norwegian University of Science and Technology where he is working to develop novel hybrid methods at the intersection of big data, physics driven modelling and data driven modelling in the context of real time automation and control. He also holds a part time senior scientist position in the Department of Mathematics and Cybernetics at SINTEF Digital where he led the Computational Sciences and Engineering group between 2012-2018. He holds a PhD in Multiscale Modeling of Urban Climate from the Swiss Federal Institute of Technology Lausanne. Prior to that he received his bachelors in Mechanical Engineering and a masters in Thermal and Fluids Engineering from the Indian Institute of Technology Bombay.



Arne Hestnes is a software architect at the Kongsberg Maritime Sensor and Robotics division. He is a Master of Technology in machine learning from the Norwegian University of Science and Technology. Currently the main work evolves around connecting industrial sensors to cloud and automating processing flows.