On the detection and classification of objects in scarce sidescan sonar image dataset with deep learning methods

Yannik Steiniger[1], Jannis Stoppe[1], Dieter Kraus[2], and Tobias Meisen[3]

[1]German Aerospace Center, Institute for the Protection of Maritime Infrastructures, 27572 Bremerhaven, Germany
[2]City University of Applied Sciences Bremen, 28199 Bremen, Germany
[3]University of Wuppertal, 42119 Wuppertal, Germany

Contact author: Yannik Steiniger, Fischkai 1, 27572 Bremerhaven, Germany, yannik.steiniger@dlr.de

Abstract: *Applying deep learning detection methods to sonar imagery is a challenging task due to the complexity of the image itself as well as the limited amount of available data. In this work, we analyze one-step and two-step setups for detection multiple different objects in sidescan sonar images. The one-step setup and the first step in the two-step setup uses standard deep learning models, like YOLOv8, to either directly locate and classify the objects or to serve as a snippet extractor. In the second step these extracted snippets are further classified by a convolutional neural network. Furthermore, we investigate a setup in which the detected objects from the one-step approach are filtered by another CNN to reduce false alarms. Finally, we compare the performance of multiple deep learning detectors to a classical two-step approach using template matching combined with a CNN. Our results show that both two-step setups generate less false alarms. Furthermore, all deep learning models outperform the template matching approach.*

Keywords: *Deep Learning, Automatic Target Recognition, Sonar Imagery*

1. INTRODUCTION

While the past years have shown that deep learning methods like convolutional neural networks (CNN) can achieve excellent results in classifying sonar images [1], less research has been done regarding the deep learning based detection of objects in sonar images [2]. In classical automatic target recognition [3], first regions of interest are localized inside the sonar image. The corresponding snippets are then filtered to reduce false alarms. Finally, a classification is carried out to distinguish different objects. In contrast to this, deep learning based detectors directly combine the localization and classification into a single model. However, modern deep learning detectors rely on large training datasets, which is a critical aspect, regarding sonar imagery, as training data in this context is scarce. Splitting the detection task into localization and classification results in two models which need to be trained on less complex subtasks. It is expected, that this complexity reduction is beneficial when training deep learning models in use-cases with scarce data.

In this work, we analyze different one-step and two-step detection setups, as shown in Figure 1. More precisely, we use the deep learning methods YOLOv3 [4], YOLOv8 [5] and CenterNet2 [6] to detect and classify different objects in sidescan sonar images in a singe step. Furthermore, for our two-step approach, these models are trained to detect all objects as general targets. In the second step a CNN is used to carried out the classification. Another two-step setup uses a binary classifier CNN to filter false alarms from the one-step detectors. Our results show that the two-step setup generates less false alarms at the cost of a slightly lower maximum true positive rate. Furthermore, we compare the deep learning based detection to a classical template matching approach [7] and show that all evaluated deep learning models outperform the combination of template matching and CNN.

2. METHODS AND MATERIALS

2.1. DETECTION METHODS

As mentioned before, we investigate the three deep learning detectors YOLOv3, YOLOv8 and CenterNet2 as well as a template matching approach. YOLOv3 is a common one-stage deep learning detector which was already found to be suitable for detecting objects in sonar images [8]. For a given input image, YOLOv3 predicts the location, size and class of present objects. The predicted class of the object is the result from multiplying an objectness score with a class probability. The location and size are expressed in terms of a rectangular bounding box, which is regressed relative to so-called anchors or bounding box priors. Because multiple anchors are considered, non-maximum suppression is applied to the predicted bounding boxes to remove multiple detections of the same object. The backbone network of YOLOv3, which extracts features from the input image, is the CNN Darknet-53. Feature maps from different depth of Darknet-53 are used to detect objects at different scales which has shown to improve the detection of smaller objects [4].

Several extensions and updates of the YOLO architecture were proposed during the past few years. Very recently, YOLOv8 was published by Ultralytics [5]. It builds on the CSPDarknet-53 backbone introduced in YOLOv4 [9]. In contrast to most previous YOLO versions, YOLOv8 directly predicts a class without an objectness score. Furthermore, the bounding box is predicted

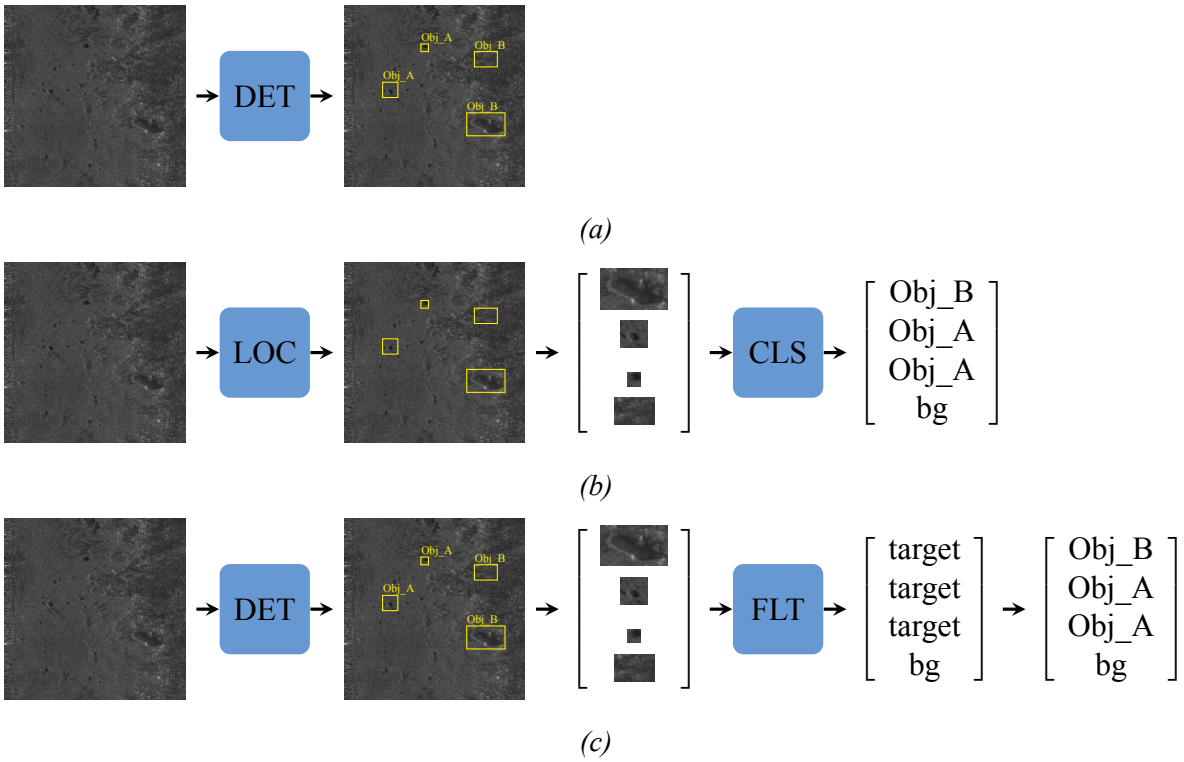


Figure 1: Experimental setups. (a) The detector (DET) localizes and classifies different objects in a waterfall image. (b) The detector is only used for localization (LOC) while a CNN classifies the extracted snippets in a subsequent step (CLS). (c) The detector is used as in (a) and another CNN carries out a binary classification (FLT) to reduce the number of false alarms by filtering background snippets.

as the distance from a fixed anchor point to each side of the bounding box. This anchor-free detection eliminates the need for filtering the bounding boxes by non-maximum suppression. For a more detailed overview about the evolution of the YOLO architectures see [10].

CenterNet2 is a so-called probabilistic two-stage detector. Here a one-stage detector is used as the region proposal network. Rather than producing a large number of proposals to ensure a high recall, the aim of the first stage is to generate boxes with a high objectness score, i.e., a high likelihood that the box contains an object. The subsequent second stage then performs the final bounding box regression and classification. This approach is very similar to our two-step setup, however, here we use this two-*stage* detector as the first *step* in our setup. CenterNet2 uses CenterNet [11] as the first-stage detector and Cascade-RCNN [12] in the second stage. Unlike YOLOv8, the detection of CenterNet is based on keypoints instead of anchor points. These keypoints are predicted by the network and determine the center of the object, from which subsequently the width and height of the box is predicted.

Finally, to be able to compare the deep learning based detection to a classical approach, we implement a template matcher [7]. Correlating a template with the input sonar image generates a heatmap in which high values indicate the presence of an object. After thresholding this heatmap a fixed size bounding box, which size is defined by the size of the template, is used for locating the object. In total 25 templates are generated covering different object shapes, sizes, rotations and shadow length.

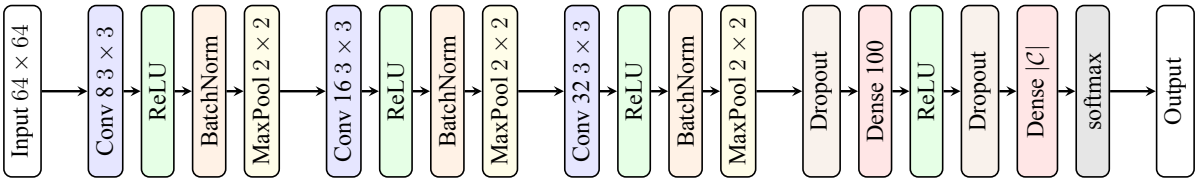


Figure 2: Architecture of the classifier CNN. $|\mathcal{C}|$ is set to 5 for the multi-class and to 2 for binary classification.

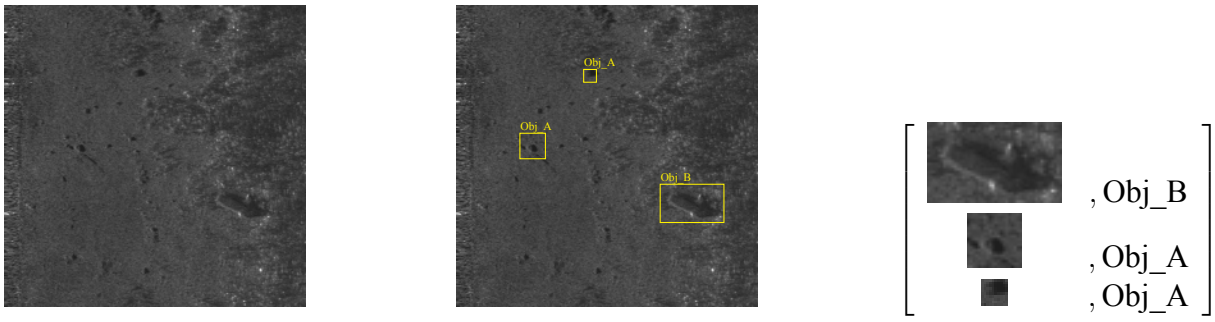
2.2. CLASSIFICATION METHODS

In our two-step approach we use a CNN to classify the regions of interest from the first localization step. The CNN, which we introduced in a previous work [13], follows the conceptual approach that the number of kernels are doubled once the dimension of the feature map is halved [14]. Figure 2 shows the architecture of this CNN. When used for multi-class classification the number of neurons in the output layer is set to the number of detectable classes (here $|\mathcal{C}| = 5$). For the binary classification between general targets and false alarms another CNN with the same structure but two output neurons is trained, instead.

2.3. SIDESCAN SONAR DATASET

We need two datasets in this work: a detection and a classification dataset. In general we use sidescan sonar data presented and described in detail in our previous work [8, 13]. Grayscaled waterfall images with a resolution of 10 cm in along and in across track direction are formed from the raw sonar data. An example is shown in Figure 3a. In these waterfall images we labeled objects from the four classes *Tire*, *Rock*, *Cylinder* and *Wreck* with manually defined bounding boxes tightly enclosing the object highlight and acoustic shadow (see Figure 3b). The detection dataset consists of the whole waterfall images as input and the bounding box coordinates and class labels as target variables. For the classification dataset the snippets defined by the bounding boxes are extracted from the waterfall image. In addition, random background snippets are selected to form the fifth class, which we refer to as *Background*. The final classification dataset consists of the extracted snippets as input and the class labels as the target variable (see Figure 3c).

Images from the classes *Tire*, *Cylinder* and *Wreck* are very limited. Thus, the training and test split for both datasets is done such that these classes are nearly split 50:50. However, multiple images of the same objects, e.g., from different viewing angels, exist in the dataset. We split the data such that images from the same object are either in the training or test set. Additionally, if multiple objects are in the same waterfall image, their associated snippets are all assigned to either the training or test set. This way we ensure that the same objects are used for training and testing in the detection and in the classification task. These restrictions result in the number of samples in the training and test set for each class reported in Table 1.



(a) *(b)* *(c)*

Figure 3: (a) Example of a waterfall image. (b) Waterfall image labeled for detection. (c) Extracted snippets and corresponding class labels for classification.

Table 1: Overview about the datasets. Waterfall images are used for detection. Snippets from the classes Tire, Rock, Cylinder, Wreck and Background are used for classification.

	Number of					
Dataset	**waterfall images**	**tires**	**rocks**	**cylinders**	**wrecks**	**background**
Training	769	24	2288	15	10	1390
Test	128	12	167	22	10	719

3. EXPERIMENTAL SETUP

Three different detection setups are analyzed in this work, as previously illustrated in Figure 1. First, as in standard deep learning detection, one model is used to localize and classify objects in the sonar image. In the experiments, the general block DET in Figure 1 is substituted by each of the three deep learning models YOLOv3, YOLOv8 and CenterNet2. Secondly, the detector is only used for localizing objects (LOC) in the image. Here both conventional, i.e. template matching, as well as deep learning based detectors are used. When training the deep learning detectors in this setup, all objects are considered as a single general class *Target*. In the following, these models are named YOLOv3-L, YOLOv8-L and CenterNet2-L. The snippets extracted by the predicted bounding boxes are classified in the next step by a CNN (CLS). This CNN is trained on snippets from the classification training dataset (see Table 1). Finally, the third setup extends the first one by adding another CNN which carries out a binary classification between *Target* and *Background* (FLT). Similar to the LOC training, when training this CNN all object classes are combined into *Target*. Detected objects from the first step are filtered if the CNN in the second step classifies the snippet as *Background*. Otherwise, the class predicted by the detector is assigned to the snippet.

Following common practice, all deep learning detectors, also if used for localization only, are pre-trained on the MS COCO dataset. When trained on the sonar data, the images from the port-side sonar are flipped such that the shadow of an object always lies on the right side of the object which enforces invariance to this variability. During training, the dataset is augmented using horizontal flipping only, since vertical flipping would cause the shadow to lie on the left side of the object again. We use YOLOv8 and CenterNet2 in their standard configuration and YOLOv3 as described in our previous research [8]. All detectors are trained for 100 epochs on the waterfall images. We found that due to the limited amount of training data the detection score of multiple true objects is very small. Thus, at inference, we set the detection threshold to 0.0

Method	TPR_{max}	**False alarms @** TPR_{max}	**False alarms @** TPR**=0.9**
YOLOv3	**1.000**	196.814	159.543
YOLOv8	0.957	196.884	168.907
CenterNet2	0.934	196.922	93.233
YOLOv3-L+CNN	0.900	**12.636**	**12.636**
YOLOv8-L+CNN	0.934	57.333	47.372
CenterNet2-L+CNN	0.853	22.961	-
TM+CNN	0.616	31.295	-
YOLOv3+CNN	0.981	56.744	47.566
YOLOv8+CNN	0.938	55.372	42.310
CenterNet2+CNN	0.863	45.628	-

Table 2: Maximum true positive rate and number of false alarms for the methods with d=20 px.

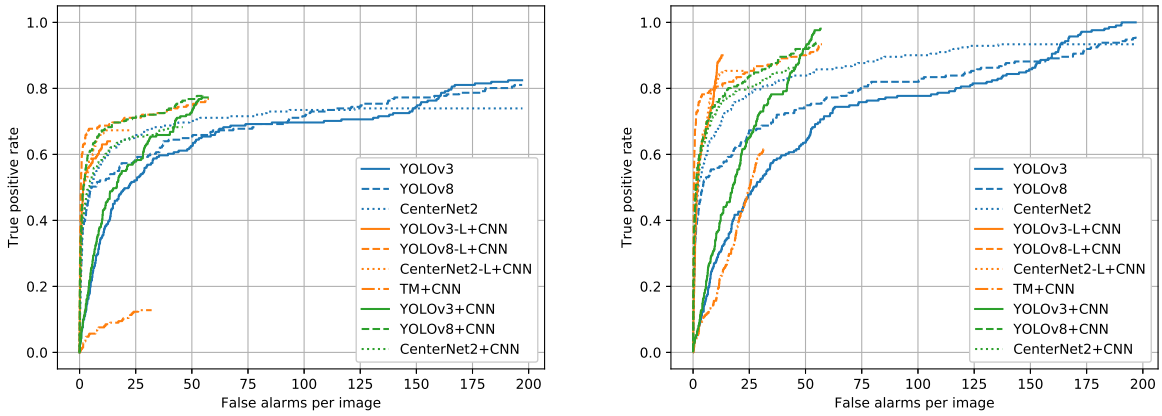
and limit the number of detections to 200 to ensure a high recall. The multi-class and binary classifier CNNs are trained for 100 epochs with the Adam optimizer. The initial learning rate is set to 0.0001 and reduced to 0.00001 after the first 50 epochs. According to [15] horizontal flipping, cropping and the addition of Gaussian noise is used for augmentation.

The detection performance in all setups is compared by means of ROC-like curves. Note that for detection the false positive rate cannot be calculated because there are no true negative cases. Thus, we display the average number of false alarms per image on the abscissa and refer to these curves as ROC-like. The curves are generated by varying the detection threshold for the confidence score of the detectors and counting the number of true positive detections and false alarms in all test images. Since the template matcher only predicts bounding boxes of fixed sizes, not only the intersection over union (IoU) is considered to determine true positive detections but also the pixel-wise Euclidean distance d between the center pixel of the true and predicted bounding box.

4. RESULTS

Figure 4 displays the ROC-like curves for the three setups described above. As expected, adding the binary classifier CNN to the deep learning detectors reduces the false alarm rate. An even lower false alarm rate is achieved if the detectors are only used for localization and the CNN classifies the snippets. Only for YOLOv8 the performance of YOLOv8-L+CNN and YOLOv8+CNN is similar. For all deep learning detectors, the maximum true positive rate (TPR_{max}) is lower in both two-step setups than in the one-step approach. Table 2 lists TPR_{max}, the false alarm rate corresponding to this value as well as the false alarm rate for a true positive rate of 0.9. Here the true positives are determined using $d = 20$ px. Interestingly, YOLOv3 is the only model achieving a true positive rate of 1. In addition, YOLOv3-L+CNN is the method with the lowest false alarm rate at a high true positive rate. This result surprises since YOLOv8 and CenterNet2 outperform YOLOv3 on standard computer vision benchmarks like MS COCO.

Comparing the individual detectors, CenterNet2 performs best in the one-step setting for low true positive rates but lacks the capability to detect all targets. Looking at the distance-based true positive measure, when comparing YOLOv3 and YOLOv8, a turning point can be seen in all three setups, before which YOLOv8 achieve better results. However, YOLOv8 struggles to detect all objects leading to a flattening of the ROC-like curve and finally a better performance



(a) *(b)*

Figure 4: ROC-like curves with (a) IoU=0.5 and (b) d=20 px.

of YOLOv3. Nevertheless, when the IoU is used to determine a true positive detection YOLOv8 is the best detector in the two-step setups. This indicates that the bounding boxes predicted by YOLOv8 fit better to the specific objects. Finally, all deep learning methods outperform the template matcher in this setup by a large margin.

For a further inside into the detection performance, Figure 5 shows the detections of the individual methods in the two-step setup LOC+CLS for two example images. Here the detection threshold is set to 0.1. The deep learning detectors clearly generate less false alarms than the template matcher. Note however, that the detection threshold for the template matcher is related to the correlation and cannot directly be compared with the ones from the deep learning models. YOLOv3 is not able to separate the two close rocks in the first example. YOLOv8 and CenterNet2 both detect them as two objects but at the same time generate more false alarms. For most false alarms of the deep learning models, a highlight-shadow structure, e.g., caused by small hills, is visible and responsible for the detection. An additional fine-tuning of the classifier CNN using such snippets as *Background* could reduce false alarms even further. The template matcher itself is very active in areas containing objects and in darker regions of the image. However, the placement and size of the bounding boxes is not accurate enough to extract snippets which can be properly classified by the CNN. Thus, many detections are filtered out as background. A network which is more invariant to translations of the object inside the snippet could increase the performance of this method.

5. SUMMARY

In this paper we have investigated one-step and two-step detection setups for detecting different objects in sidescan sonar images. The deep learning detectors YOLOv3, YOLOv8 and CenterNet2 were studied. All detectors achieved the highest true positive rate in the one-step setup, however, at the cost of a high false alarm rate. With a two-step setup the number of false alarms could be reduced by a large amount. The best performance is achieved with YOLOv3-L+CNN, which generates 12.6 false alarms per image at a true positive rate of 90%. All deep learning models outperform a classical template matcher. Further analysis should consider more sophisticated detection algorithms like the Mondrian-detector [16] or its updated version [17]. In addition, a broader comparison of different deep learning methods, also taking transformer

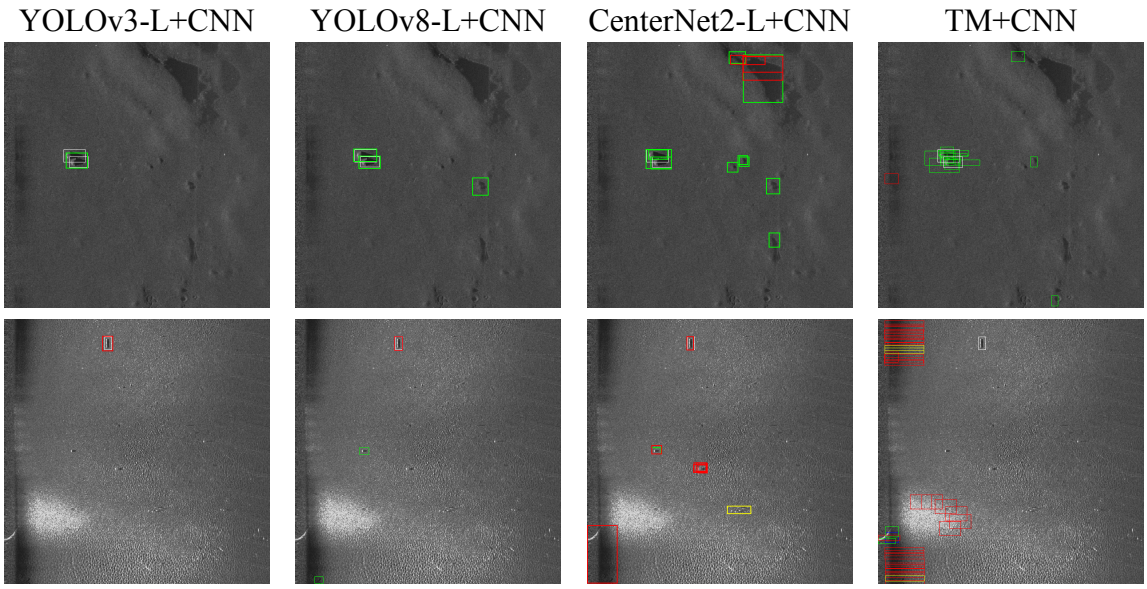


Figure 5: Example detections. Bounding box color indicates object class as: green - rock, red - cylinder, yellow - boat, white - ground truth

based approaches like DETR or SWIN into account, is a next step to find the method most suited for analyzing sonar images. This becomes more relevant since our results show that newer and larger deep learning models not necessary perform better on sonar images due to the limited amount of available data.

REFERENCES

[1] D. P. Williams: "On the use of tiny convolutional neural networks for human-expert-level classification performance in sonar imagery", *IEEE Journal of Oceanic Engineering* **46** (1), 236–260 **(2021)**.

[2] Y. Steiniger, D. Kraus, T. Meisen: "Survey on deep learning based computer vision for sonar imagery", *Engineering Applications of Artificial Intelligence* **114**, 105157 **(2022)**.

[3] T. Fei, D. Kraus, A. M. Zoubir: "Contributions to Automatic Target Recognition Systems for Underwater Mine Classification", IEEE Transactions on Geoscience and Remote Sensing **53** (1), 505-518 **(2015)**.

[4] J. Redmon, A. Farhadi: "YOLOv3: An Incremental Improvement", *arXiv:1804.02767*, 2018, [online] Available: https://arxiv.org/abs/1804.02767.

[5] G. Jocher, A. Chaurasia, J. Qiu: "YOLO by Ultralytics" (8.0.0), 2023, [Software] https://github.com/ultralytics/ultralytics.

[6] X. Zhou, V. Koltun, P. Krähenbühl: "Probabilistic two-stage detection", *arXiv:2103.07461*, 2018, [online] Available: https://arxiv.org/abs/2103.07461.

[7] B. Lehmann, K. Siantidis, I. Aleksi, D. Kraus: "Efficient Pre-Segmentation Algorithm for Sidescan-Sonar Images" in *7th International Symposium on Image and Signal Processing and Analysis* (Dubrovnik, 2011).

[8] Y. Steiniger, J. Groen, J. Stoppe, D. Kraus, T. Meisen: “A study on modern deep learning detection algorithms for automatic target recognition in sidescan sonar images”, in *6th Underwater Acoustics Conference and Exhibition* (Virtual, 2021).

[9] A. Bochkovskiy, C.-Y. Wang, H.-Y. M. Liao: “YOLOv4: Optimal Speed and Accuracy of Object Detection”, *arxiv:2004.10934*, 2020, [online] Available: https://arxiv.org/abs/2004.10934.

[10] J. R. Terven, D. M. Cordova-Esparaza: “A Comprehensive Review of YOLO: From YOLOv1 to YOLOv8 and Beyond”, *arXiv:2304.00501*, 2023, [online] Available: https://arxiv.org/abs/2304.00501.

[11] X. Zhou, D. Wang, P. Krähenbühl: “Objects as Points”, *arXiv:1904.07850*, 2019, [online] Available: https://arxiv.org/abs/1904.07850.

[12] Z. Cai, N. Vasconcelos: “Cascade R-CNN: Delving into High Quality Object Detection” in *IEEE Conference on Computer Vision and Pattern Recognition* (Salt Lake City, 2018).

[13] Y. Steiniger, A. Bueno, D. Kraus, T. Meisen: “Tackling data scarcity in sonar image classification with hybrid scattering neural networks”, in *OCEANS 2023, Limerick* (Limerick, 2023).

[14] K. Simonyan, A. Zisserman: “Very Deep Convolutional Networks for Large-Scale Image Recognition” in *3rd International Conference on Learning Representations*, (San Diego, 2015)

[15] Y. Steiniger, J. Stoppe, D. Kraus, T. Meisen: “Investigating the training of convolutional neural networks with limited sidescan sonar image datasets” in *OCEANS 2022, Hampton Roads* (Hampton Roads, 2022).

[16] D. P. Williams: “The Mondrian detection algorithm for sonar imagery”, *IEEE Transactions on Geoscience and Remote Sensing* **56** (2), 1091-1102 **(2018)**.

[17] D. P. Williams: “On the Utility of Multiple Sonar Imaging Bands for Underwater Object Recognition” in *OCEANS 2022, Hampton Roads* (Hampton Roads, 2022).