From TrashCan to UNO: Deriving an Underwater Image Dataset To Get a More Consistent and Balanced Version

Cyril Barrelet[1][0000−0003−4413−6885], Marc Chaumont[1,3][0000−0002−4095−4410], Gérard Subsol[1][0000−0002−7461−4932], Vincent Creuze[1][0000−0002−6813−8562], and Marc Gouttefarde[2][0000−0002−4369−6867]

[1] Research-team ICAR, LIRMM, Univ Montpellier, CNRS, Montpellier, France
[2] Research-team DEXTER, LIRMM, Univ Montpellier, CNRS, Montpellier, France
[3] Univ Nîmes, France
`cyril.barrelet@lirmm.fr`

Abstract. The multiplication of publicly available datasets makes it possible to develop Deep Learning models for many real-world applications. However, some domains are still poorly explored, and their related datasets are often small or inconsistent. In addition, some biases linked to the dataset construction or labeling may give the impression that a model is particularly efficient. Therefore, evaluating a model requires a clear understanding of the database. Moreover, a model often reflects a given dataset's performance and may deteriorate if a shift exists between the training dataset and real-world data.
In this paper, we derive a more consistent and balanced version of the TrashCan [6] image dataset, called UNO, to evaluate models for detecting non-natural objects in the underwater environment. We propose a method to balance the number of annotations and images for cross-evaluation. We then compare the performance of a SOTA object detection model when using TrashCAN and UNO datasets. Additionally, we assess covariate shift by testing the model on an image dataset for real-world application. Experimental results show significantly better and more consistent performance using the UNO dataset.
The UNO database and the code are publicly available at:
https://www.lirmm.fr/uno and
https://github.com/CBarrelet/balanced_kfold.

Keywords: underwater imagery, underwater trash, inbalanced dataset, Deep Learning

1 Introduction

Recently, interest in cleaning up the seabed has increased, motivated by the growth of underwater macro-litter pollution [2]. Monitoring the presence of macro-litter on the seabed, in particular by optical acquisition, has become a very active research topic [10]. Deep Learning (DL) approaches are then adapted to detect, localize, and identify macro-litter in the underwater environment, which is very variable. Moreover, these approaches should run fast, which is crucial if we want to guide Remotely Operated underwater Vehicles (ROV) to pick macro-litter.

The availability of video datasets as DeepSeaWaste [5][4] or TrashCan [7][5] allowed researchers to design DL networks to classify [16] [13] and localize [11] [9] macro-litter within underwater images. Figure 1 shows some images from these two datasets.

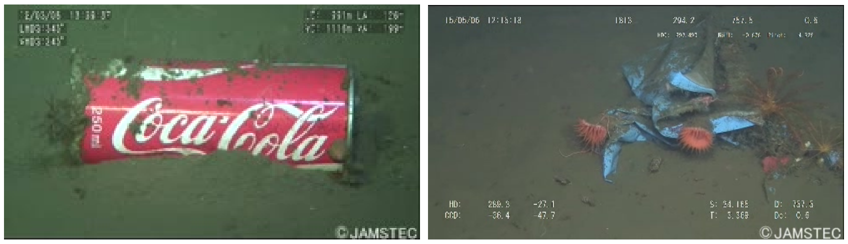

Fig. 1. Images from the DeepSeaWaste (left) and TrashCan (right) datasets.

The DeepSeaWaste dataset [5] is composed of 544 underwater macro-litter images taken by the Japan Agency of Marine-Earth Science and Technology (JAMSTEC) [1]. The 76 classes are descriptive, such as *ashtray*, *bottle*, or *rope*, but they are entirely unbalanced, considering that the number of examples ranges from 1 to 12. In fact, 43 classes have only two examples at most. Moreover, some images have multiple labels. The limited number of images for some classes makes the training or evaluation of classification tasks difficult.

The TrashCan dataset [7] is a semantically-segmented database composed of 7,212 images that were primarily extracted consecutively from 312 different video sequences taken by JAMSTEC, since 1982, in the Sea of Japan. It is an improved version of the Trash-ICRA19 dataset [4]. 16 to 22 classes are represented depending on the version, such as *ROV* (which includes any part of the ROV carrying the camera), animals (which are defined by specific classes such as *animal_ crab*, *animal_ eel*, *animal_fish*), plant, and trash which is also divided into specific subclasses such as *trash_ metal* or *trash_fishing_ gear*. However, the classes are poorly balanced, particularly the *ROV* category, which represents up to 33% of all annotations. We also visually assessed that many annotations are incorrect, poorly localized, or missing. Notice that some metadata, such as the depth, date, or time, are directly overlayed on the images, which could introduce some artifacts in the learning process.

[4] https://www.kaggle.com/henryhaefliger/deepseawaste
[5] https://conservancy.umn.edu/handle/11299/214865

Another problem occurs in the evaluation. When we use the standard k-fold cross-validation method, we split the dataset into k sub-datasets, and, for each fold, we keep *k-1* sub-datasets for constituting the training dataset, and the last one becomes a test dataset. Thus, we get k detection/localization accuracies that we average to estimate an overall accuracy. In the TrashCan dataset, the 7,212 images were extracted consecutively from videos sequences, so 2 images might be very similar if they belong to the same sequence, especially if the ROV is moving very slowly or is stable. Consequently, if in the k-fold evaluation process, we randomly distribute the images over the k sub-datasets, we will have images of a given sequence both in the training and the test datasets. As these images may be very close, they may artificially boost the overall performance.

In conclusion, there are very few available underwater macro-litter datasets, and they suffer from a strong imbalance between classes, many annotation errors, and temporal consistency, which make many images of the dataset very close. In this work, we propose to take the TrashCan dataset, which is the most complete and relevant one for macro-litter detection and localization in the underwater environment, and derive a new version that will be more consistent and balanced. In the following, we describe our method. Notice that even if the methodology is focused on a specific dataset, it could be used to derive other image datasets presenting the same limitations.

The rest of the paper is organized as follows. Section II describes the deriving process itself (frame selection, class fusion, text suppression, and re-labeling) in order to obtain a new dataset called UNO (for Underwater Non-natural Object), which is publicly available. Section III describes, in particular, a methodology (available on the GitHub[6]) to get k-folds, ensuring that images of the same video are only in one training or evaluation dataset, thereby resulting in a good balance of frames and bounding boxes. In Section IV, the state-of-the-art YOLOv5 [9] detector is used to compare performances using TrashCan and its derived version UNO as training or evaluation datasets. We also evaluate the effect of using TrashCan or UNO as the training dataset on the generalization ability by evaluating performances on images of the AQUALOC dataset [3], which presents a significant covariate shift.

2 UNO dataset construction

2.1 Label redefinition

Our objective in using the TrashCan dataset is to develop methods to detect or localize macro-litter, but the dataset also includes non-litter classes such as *animal*, *plant*, or *ROV*. The trash class itself is decomposed into 8 distinct classes with 142 to 2,040 examples per class. In order to mitigate class imbalance and the ambiguity of trash definition, we decided to consider a more general problem

[6] https://github.com/CBarrelet/balanced_kfold

of object localization by fusing all the trash classes as the *ROV* one into a unique class that we call *Non-natural Object*. This postpones the classification step to future work.

2.2 Text removal

As we can see in Figures 1 and 2, some metadata are directly overlayed as text in the images. Therefore, we must remove it as it may disturb the detection process in the learning or test phase if the text is considered as an object.

More precisely, the text is always on the top and bottom parts of the image. We thus decided to suppress the top and the bottom of the frame leading to cropped images with smaller heights. To this end, we set up an automatic processing consisting in first, detecting the text, secondly estimating the average ordinate (y-axis) for the top and the bottom text line, and third cropping the middle region of the image (see Figure 2, column 2). More practically, we train a YOLOv5 extra-large model for text detection on the COCO-Text dataset [15] with the recommended hyperparameters from [9] and a batch size of 28, achieving a precision (P) of 0.71, a recall (R) of 0.56, and a mAP@.5 of 0.621. Since the precision remains relatively low, we keep only the detected text which y coordinate is within the mean $\pm$ std of all y coordinates of the detected texts. In fact, the detected text laying outside this interval is considered as false positive and then removed.

2.3 Relocalization

TrashCan has a non-negligible part of the labels that either are incorrect, missing, or which corresponding bounding boxes (BB) are imprecisely located. We relocated every BB w.r.t. the following rules. First, we modify the border delimitation between objects and bounding boxes, assuring a perfect pixel tightness. We removed all misplaced BB and tried to avoid overlapping BB that might cause a performance drop. Secondly, we annotated missing objects from frame to frame. Furthermore, we chose to add some complex examples. Indeed, objects are barely recognizable from a human perspective because of depth, luminosity, and turbidity, in the underwater environment. So for a labeled object in a given frame, we checked for the same object in the previous frames even if its appearance is unclear and added the frame to the dataset.

2.4 Discussion on the derived UNO dataset

In Figure 2, we can see in the TrashCan images that the ROV (up), which partly appears on the bottom and the upper right of the image is delineated by two different BB which are much larger than the ROV parts and (bottom) that the plastic bottle is not labeled. The UNO images are cropped parts of the TrashCan images in order to delete non-significant content as the text and the BBs were resized in order to better fit the non-natural objects.

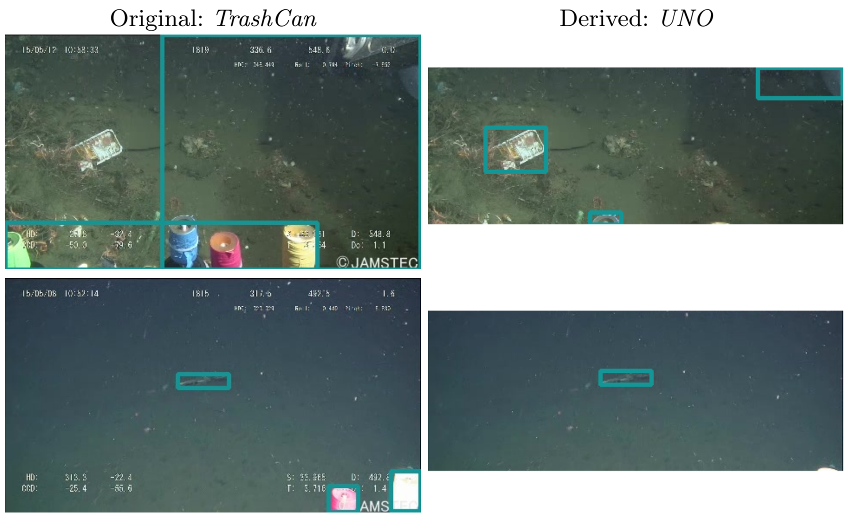

Fig. 2. Images and annotations from TrashCan (left) and UNO (right) datasets. We can see how the UNO images were cropped in order to delete artefactual content (in particular the text) and how some BB were resized or added to fit better the non-natural objects.

After the process, the UNO dataset consists of 279 video sequences with 5,902 frames and 10,773 bounding boxes labeled with the unique Non-natural object class.

We pointed out some limitations directly linked to the localization and the shape of the objects. We included some overlapping BBs while minimizing their number to limit confusion at evaluation time. Moreover, thin diagonally shaped objects, such as ropes, could confuse the model at the training stage. Indeed, a thin diagonally shaped object covers just a small portion of its BB, while the background covers the rest. We could have used polygonal BB and pixel-wise labels to improve diagonally-shaped object detection. Nevertheless, we labeled them with classic rectangle BB because of their rare occurrence.

3 A methodology for a well-balanced k-fold

When a dataset contains a small number of images, the best way to properly evaluate the accuracy of a DL network is to split the dataset in k-folds and circularly use *(k-1)* folds as the learning dataset and the last fold for the test dataset. This results in a total of k learning phases, each evaluated on a different fold. Finally, one can average all the results in an overall performance indicator (i.e., accuracy) and compute the standard deviation. This last value can be considered as an indicator of generalizability and can be used in a Student's t-test on the performance indicator to decide if a network is better than another.

Note that even if one wants to split the dataset only into 2 folds (a training set and a test set), one still must be cautious. In the case of TrashCan, a direct random split introduces a bias. As mentioned, the frames of the dataset are extracted consecutively from the videos. A frame n and $n+1$ of a video might be very similar, and they may be assigned one in the training set and the other in the test set. As they are very close, they may artificially boost the model performance. Therefore, the correct way to split the dataset is to group all the frames belonging to one video into a unique fold. Figure 3 shows the number of frames per video and its variability.

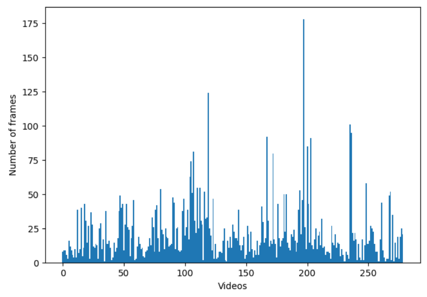

Fig. 3. Number of frames for the 279 videos.

Moreover, in order to get comparable learning phases in the cross-validation process, we must ensure that we have the same number of BB in each fold. In conclusion, we have to split the dataset into k-folds (in the following, we fixed k to 5) with both balanced frames and BB numbers, while keeping all the frames of a video in a unique fold. This problem is known as the bin packing problem [12] since we want to fill $k = 5$ folds at the best (with the video frame constraint), knowing their targeted capacity is approximately the total number of frames and BB divided by 5.

As we are looking for approximately the same number of frames and the same number of BB in each frame, we will minimize both the standard deviation σ_F of the number of frames per fold and the standard deviation σ_{BB} of the number of BB. Due to the pertinence of both the frames and the BB numbers, we choose to consider σ_F and σ_{BB} equally. Thus, the problem can then be written as the following optimization:

$$f^* = arg \min_{f \in \{1..5\}^{279}} (\sigma_F + \sigma_{BB}) \tag{1}$$

With f a 279-tuple, i.e., $f \in \{1, ..., 5\}^{279}$, corresponding to the assignment of 279 videos to 1 of the 5 folds. There are 5^{279} different 279-tuples which is very large.

An approximate but fast and easy solution to find the best assignment f is to create 100,000,000 k-folds by shuffling the videos and filling each fold to their targeted capacity. Then, we select the k-fold assignment minimizing equation 1. Table 1 shows the distribution obtained in our best k-fold. The result seems reasonable since both σ_F and σ_{BB} are very low.

Table 1. Optimal 5-fold distribution with video, frame and BB numbers by fold.

Fold	Video	Frames	BBs
1	63	1180	2159
2	64	1182	2137
3	49	1185	2152
4	44	1179	2163
5	57	1176	2162
Mean	55.4	1189.2	2154.6
σ	7.81	3.00	9.60

More generally, since a different video database could lead to a scattered distribution, one may privilege a standard deviation over another in favor of a more balanced version. Hence, the problem could be rewritten as the following optimization:

$$f^* = arg \underset{f \in \{1..5\}^N}{min} \Big((1-\alpha)\,\sigma_F + \alpha\,\sigma_{BB} \Big) \tag{2}$$

With f a N-tuple, i.e., $f \in \{1, ..., 5\}^N$, corresponding to the assignment of N videos to 1 of the 5 folds, and $\alpha \in [0, ..., 1]$ a weighting parameter.

4 Experiments and results

In this section, the UNO dataset is benchmarked with YOLOv5m [9]. As a recall, the UNO dataset contains 279 videos, 5,902 frames, and 10,773 bounding boxes. All bounding boxes contain a non-natural object; YOLOv5m is then a one-class detector.

4.1 Experiments

We chose the YOLOv5m [9] model pre-trained on ImageNet at a 640×640 pixels resolution. We used transfer learning from its pre-trained weights to keep the previous knowledge. We chose the SGD optimizer and the OneCycle scheduler, with initial and final learning rates of 0.0032 and 0.000384, respectively, while setting the warmup at 20% of the total epochs. In addition, we set the batch

Table 2. Results on different dataset achieved using YOLOv5m

Training set	**Evaluation set**	**Split**	**P (%)**	**R (%)**	**F1 (%)**	**mAP@.5 (%)**
TrashCan	TrashCan	Random	83.1	76.5	79.7	80.8
TrashCan	TrashCan	K-folded	57.1 ± 5.2	60.1 ± 5.5	58.4 ± 4.2	56.6 ± 6.3
TrashCan	UNO	K-folded	64.2 ± 5.0	58.2 ± 2.9	60.9 ± 2.6	60.8 ± 4.2
UNO	**UNO**	**K-folded**	**68.7 ± 4.3**	**66.2 ± 1.5**	**67.3 ± 1.5**	**68.8 ± 1.2**

Table 3. Covariate shift results using YOLOv5m

Training set	**Evaluation set**	**Split**	**P (%)**	**R (%)**	**F1 (%)**	**mAP@.5 (%)**
TrashCan	AquaLoc	K-folded	59.8 ± 6.3	52.9 ± 3.8	55.7 ± 1.6	52.5 ± 1.9
UNO	**AquaLoc**	**K-folded**	**61.2 ± 3.1**	**51.2 ± 6.2**	**55.6 ± 4.5**	**55.2 ± 4.7**

size to 28, which is the maximum according to our GPU capability (NVIDIA[7] Quadro RTX 6000 with 24Go VRAM). We set the IoU threshold to 0.2 between predictions and labels for evaluation. We also used data augmentation given in the YOLOv5 fine-tuning V5.0 version, such as color transformation, rotations, translations, scaling, shearing, flip-UP, flip-LR, mosaic, and mixup.

As mentioned, we ran five training of 300 epochs each and five tests for every experiment. Each experiment took approximately seven hours.

4.2 Results

UNO benchmark and improvement with respect to TrashCan Table 2 shows the results of various experiments with the same settings and k-fold to compare TrashCan and UNO databases.

In the first row, we evaluate the results without k-folding, using a regular random split to show the bias linked to datasets containing consecutive frames. The results are biased because both the training and validation sets contain similar frames, leading to excellent results (P=83.1%, R=76.5%, F1=79.7%, and mAP@0.5=80.8%) but an erroneous conclusion. Then, we use a correct distribution preventing consecutive frames to be in both training and validation sets, i.e. using our k-fold methodology. Thus, the following rows indicate the correct evaluation, where P, R, and mAP@.5 are below 70%.

In the second row, we evaluate the TrashCan model using our k-fold methodology, resulting in P=57.1 ± 5.2%, R=60.1 ± 5.5%, F1=58.4 ± 4.2%, and mAP@.5=56.6 ± 6.3%, whereas in the third row, we evaluate the same model but on the UNO validation split to see its performance on a cleaner version, resulting in P=64.2 ± 5.0%, R=58.2 ± 2.9%, F1=60.9 ± 2.6%, and mAP@.5=60.8 ± 4.2%. The latter shows an overall improvement indicating the importance of a clean evaluation set.

[7] https://www.nvidia.com/

Finally, we give the UNO model evaluation on its validation set in the fourth row. The results increase by almost 10 points (P=68.7 ± 4.3%, R=66.2 ± 1.5%, F1=67.3 ± 1.5%, and mAP@.5=68.8 ± 1.2%). Moreover, when trained on the UNO database, YOLOv5 obtained much lower standard deviation results. UNO has thus better properties when it comes to comparing networks.

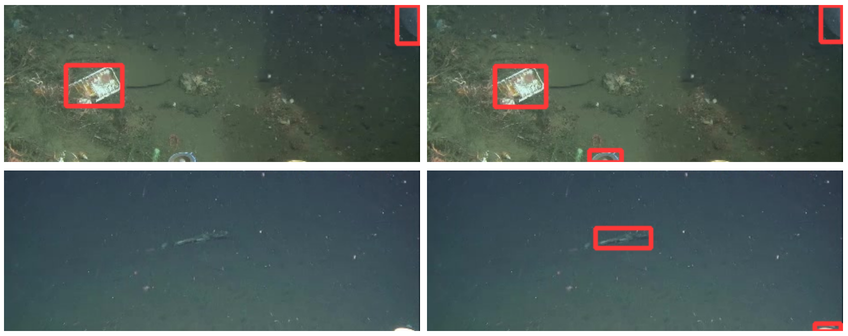

Fig. 4. Localization results on UNO images for models trained on the same subpart of TrashCan and UNO datasets, respectively.

Figure 4 gives a visual comparison of both the TrashCan and UNO models (the actual annotations are visible in Figure 2) . In the upper row, although the TrashCan model could localize the plastic bottle, it did not localize the bottom part of the ROV as the UNO model does. In the bottom row, we can see that the TrashCan model cannot localize the small piece of wood.

Covariate shift test Given that the TrashCan dataset contains videos taken in deep water since 1982, we want to test the generalization with recent images. So we labeled 150 frames from 3 videos taken from the AquaLoc dataset [3] which were acquired in the Mediterranean sea in shallow water (see Figure 5).

Given that the image's color, the lightness, the turbidity, the objects, and the environment differ a lot from those taken by JAMSTEC, we concluded that the distribution of both datasets is unlikely to be similar. Thus, we can test the generalization of a model by testing the covariate shift.

Table 3 shows both models' results on the covariate shift dataset. While we found no noticeable difference in the F1-score for both models, the UNO model obtains a higher mAP@.5 score than TrashCan. We assumed an actual improvement since we correlated the mAP@.5 results with a Student's t-test and failed to reject the NULL hypothesis (which means both models are equivalent), with a p-value greater than 0.05 (p-value = 0.49).

Even though a model could generalize with a relatively low amount of incorrect examples, the quality of the evaluation set determines its actual perfor-

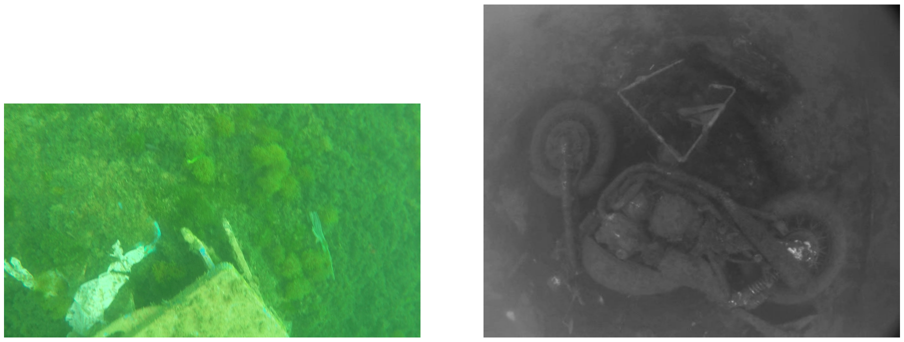

Fig. 5. Images from AquaLoc [3].

mance. Indeed, the larger the database is, the least the bad quality examples influence the overall performance. We see relatively low performances for TrashCan and UNO models regarding covariate shift testing. Indeed, JAMSTEC and AquaLoc images are very different as the color and object shapes impact the overall distribution. Moreover, a biological phenomenon called marine biological fouling makes the detection task more difficult. Marine fouling occurs when organisms grow on underwater objects. However, because it almost does not form in deep water due to the lack of light, JAMSTEC images do not cover that variability. In order to keep the gain having a pre-trained network on the UNO database but targeting use on images having a domain shift, one can envisage performing domain adaptation [14], or generative methods [6]. This is postponed to future works.

5 Conclusion

In this paper, we first selected images from TrashCan containing at least one object, set their label as a unique class, and removed the text from the images by cropping them using YOLOv5 trained on COCO-Text as a text detector. Then, we relocated every bounding box following determined rules (pixel-perfect tightness, overlapping limitation) to obtain UNO, a new dataset of underwater non-natural objects.

Secondly, we proposed a methodology (script given online) to compare networks using a well-balanced k-fold and concluded from network comparisons that UNO exhibits better properties than TrashCan.

Thirdly, we evaluated both TrashCan and UNO using YOLOv5m with the same k-fold and hyperparameters for a fair comparison.

Finally, we evaluated the learning efficiency in deploying conditions with a covariate shift test, using underwater images taken from AQUALOC for TrashCan and UNO models, and provided these images on UNO website.

As mentioned, UNO contains only one class because of the class imbalance of TrashCan. However, one could overlay this issue by creating a well-balanced underwater litter database or using few-shot classification methods [8] [17]. Moreover, the covariate shift test indicates poor detection performances for TrashCan and UNO. Because JAMSTEC images do not cover the variability of AQUALOC (shallow water, fouling, object shapes, turbidity, and luminosity), one could envision performing domain adaptation [14] or generative methods [6] to keep the gain of having a pre-trained network on the UNO database.

Acknowledgements This research has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 101000832 (Maelstrom project).

References

1. JAMSTEC: Japan Agency for Marine Earth Science and Technology, https://www.jamstec.go.jp/e/
2. Canals, M., Pham, C.K., Bergmann, M., Gutow, L., Hanke, G., Sebille, E.v., Angiolillo, M., Buhl-Mortensen, L., Cau, A., Ioakeimidis, C., Kammann, U., Lundsten, L., Papatheodorou, G., Purser, A., Sanchez-Vidal, A., Schulz, M., Vinci, M., Chiba, S., Galgani, F., Langenkämper, D., Möller, T., Nattkemper, T.W., Ruiz, M., Suikkanen, S., Woodall, L., Fakiris, E., Jack, M.E.M., Giorgetti, A.: The quest for seafloor macrolitter: a critical review of background knowledge, current methods and future prospects. Environmental Research Letters (Nov 2020). https://doi.org/10.1088/1748-9326/abc6d4, https://doi.org/10.1088/1748-9326/abc6d4, publisher: IOP Publishing
3. Ferrera, M., Creuze, V., Moras, J., Trouvé-Peloux, P.: AQUALOC: An Underwater Dataset for Visual–Inertial–Pressure Localization. Int. J. Rob. Res. **38**(14), 1549–1559 (dec 2019). https://doi.org/10.1177/0278364919883346, https://doi.org/10.1177/0278364919883346
4. Fulton, M., Hong, J., Islam, M.J., Sattar, J.: Robotic detection of marine litter using deep visual detection models. In: 2019 International Conference on Robotics and Automation (ICRA) (2019). https://doi.org/10.1109/ICRA.2019.8793975
5. Haefliger, H.: Deepseawaste (2019), www.kaggle.com/henryhaefliger/deepseawaste
6. Hong, J., Fulton, M., Sattar, J.: A generative approach towards improved robotic detection of marine litter. 2020 IEEE International Conference on Robotics and Automation (ICRA) (2020)
7. Hong, J., Michael, F., Sattar, J.: TrashCan: A Semantically-Segmented Dataset towards Visual Detection of Marine Debris. ArXiv (2020), https://conservancy.umn.edu/handle/11299/214865
8. Hu, Y., Pateux, S., Gripon, V.: Squeezing Backbone Feature Distributions to the Max for Efficient Few-Shot Learning. Algorithms (2022). https://doi.org/10.3390/a15050147, https://hal.archives-ouvertes.fr/hal-03675145
9. Jocher, G., et al.: ultralytics/yolov5: v6.0 - yolov5 (2021). https://doi.org/10.5281/zenodo.5563715
10. Madricardo, F., Ghezzo, M., Nesto, N., Mc Kiver, W.J., Faussone, G.C., Fiorin, R., Riccato, F., Mackelworth, P.C., Basta, J., De Pascalis, F., Kruss, A., Petrizzo, A., Moschino, V.: How to deal with seafloor marine litter: An overview of the state-of-the-art and future perspectives. Frontiers in Marine Science (2020). https://doi.org/10.3389/fmars.2020.505134, https://www.frontiersin.org/article/10.3389/fmars.2020.505134

11. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. IEEE Transactions on Pattern Analysis and Machine Intelligence (2015). https://doi.org/10.1109/TPAMI.2016.2577031
12. Schwerin, P., Wäscher, G.: The bin-packing problem: A problem generator and some numerical experiments with ffd packing and mtp. International Transactions in Operational Research (2006). https://doi.org/10.1111/j.1475-3995.1997.tb00093.x
13. Tan, M., Le, Q.: EfficientNet: Rethinking model scaling for convolutional neural networks. In: Chaudhuri, K., Salakhutdinov, R. (eds.) Proceedings of the 36th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 97. PMLR (2019), https://proceedings.mlr.press/v97/tan19a.html
14. Tzeng, E., Hoffman, J., Saenko, K., Darrell, T.: Adversarial discriminative domain adaptation. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE Computer Society, Los Alamitos, CA, USA (2017). https://doi.org/10.1109/CVPR.2017.316, https://doi.ieeecomputersociety.org/10.1109/CVPR.2017.316
15. Veit, A., Matera, T., Neumann, L., Matas, J., Belongie, S.: Coco-text: Dataset and benchmark for text detection and recognition in natural images (2016), https://arxiv.org/abs/1601.07140
16. Wu, H., Xin, M., Fang, W., Hu, H.M., Hu, Z.: Multi-level feature network with multi-loss for person re-identification. IEEE Access (2019). https://doi.org/10.1109/ACCESS.2019.2927052
17. Zhang, H., Cao, Z., Yan, Z., Zhang, C.: Sill-net: Feature augmentation with separated illumination representation (2021)