AI

MDPI

Article

Investigating Training Datasets of Real and Synthetic Images for Outdoor Swimmer Localisation with YOLO

Mohsen Khan Mohammadi *, Toni Schneidereit †, Ashkan Mansouri Yarahmadi † and Michael Breuß

Institute for Mathematics, BTU Cottbus-Senftenberg, Platz der Deutschen Einheit 1, 03046 Cottbus, Germany; schneton@b-tu.de (T.S.); yarahmadi@b-tu.de (A.M.Y.); breuss@b-tu.de (M.B.)

* Correspondence: khanmmoh@b-tu.de

† These authors contributed equally to this work.

Abstract: In this study, we developed and explored a methodical image augmentation technique for swimmer localisation in northern German outdoor lake environments. When it comes to enhancing swimmer safety, a main issue we have to deal with is the lack of real-world training data of such outdoor environments. Natural lighting changes, dynamic water textures, and barely visible swimming persons are key issues to address. We account for these difficulties by adopting an effective background removal technique with available training data. This allows us to edit swimmers into natural environment backgrounds for use in subsequent image augmentation. We created 17 training datasets with real images, synthetic images, and a mixture of both to investigate different aspects and characteristics of the proposed approach. The datasets were used to train YOLO architectures for possible future applications in real-time detection. The trained frameworks were then tested and evaluated on outdoor environment imagery acquired using a safety drone to investigate and confirm their usefulness for outdoor swimmer localisation.

Keywords: object detection; swimmer safety; synthetic data; background removal; YOLO architecture; image augmentation

Citation: Khan Mohammadi, M.; Schneidereit, T.; Mansouri Yarahmadi, A.; Breuß, M. Investigating Training Datasets of Real and Synthetic Images for Outdoor Swimmer Localisation with YOLO. *AI* **2024**, *5*, 576–593. https://doi.org/10.3390/ai5020030

Academic Editor: Arslan Munir

Received: 31 January 2024
Revised: 23 April 2024
Accepted: 24 April 2024
Published: 1 May 2024

1. Introduction

Swimmer safety in outdoor environments, which includes swimming pools [1], lakes [2], and seas [3], is a critical concern. One may refer to the report [4] ranking drowning as the third most significant cause of accidental injury worldwide [5], leading to approximately 320,000 individuals to lose their lives due to drowning, which accounts for 7% of all injury-related fatalities [6]. The engineering of a drowning alarm system based on computer vision incorporates several complex tasks that require, e.g., robust algorithms, suitable sensor equipment and reliable training data. Recently, drones equipped with a variety of sensors, namely visual, thermal, sonar, and also GPS systems have displayed the potential to greatly enhance swimming safety in open water environments and already act as valuable tools for lifeguards [7].

The problem we approached in this study concerns the limited availability of training data and probable complexities that a northern German outdoor lake environment may exhibit. A possible candidate for developing a robust localisation method is to make the swimmers independent from their background, which might be achieved using a background removal algorithm [8,9]. In this way, we can make use of, e.g., swimming poses not included in the available training images and therefore additionally augment our dataset.

In the domain of interest concerning our current work, a notable part of the literature focuses on feature engineering approaches. For instance, the work in [10] pursued the detection goal solely relying on colour gradient and low-level techniques, with roots in the subtraction of consecutive frames aiming to remove the swimmer background. Though this enables a fast inference, it resulted in an indoor-specific threshold-based approach with

limited generalisation ability. Let us note that the authors did not provide a model or metric for result comparisons. In [11], a video-based assistant system was developed to assist coaches to acquire swimmer positions to estimate their lap times. This was once again achieved through the utilisation of a straightforward background modelling technique followed by an engineered blob detection technique, both based on modelling the colour regime of the background water.

On the level of object detection, region-based convolutional neural networks (R-CNNs) [12–14] compute features on previously extracted region proposals to perform the object classification task. However, these so-called two-stage detectors (first step: region proposal, second step: classification) have a slow computation time. Improvements have been achieved by introducing versions of faster R-CNNs [14] and Mask R-CNNs [13]. In contrast to the R-CNN methods, YOLO (You Only Look Once) [15–20] is much faster since the underlying detection problem is approached as a regression task. A single CNN-based framework takes the entire image as an input and computes bounding boxes (localisation) and predicts class probabilities (classification) simultaneously in one stage—such methods are also referred to as one-stage detectors. Therefore, YOLO models achieve state-of-the-art computation time efficiency.

An essential component of AI-based object detection is the dataset for training and the need for a balanced representation of the to-be-learned features [21,22]. However, limited availability due to, e.g., data privacy, can undermine certain objectives. To overcome this issue, one can consider manipulating characteristics of duplicates of available images to augment a dataset [23]. In addition, to extend the number of images, augmenting a dataset may also introduce new perspectives for the detection framework, like, e.g., simulating different lighting conditions by adjusting brightness, contrast and noise, or geometric manipulations to provide the framework with additional positional information [24,25]. Another benefit of augmentation is the possible reduction in overfitting.

The overall objective of our research was the development of a supervised swimmer safety pipeline for northern German outdoor lakes as our environment of operation [26]. A major step in this process was establishing swimmer localisation [27] using AI-based object detection (YOLO), which requires training data in order to learn the features of swimming humans. However, the lack of available training data is a crucial point due to data privacy, among other things. This motivated us to utilise the existing data along with different augmentation techniques to extend the dataset for the training of the YOLO framework. Nonetheless, the unaugmented images show a limited amount of, e.g., poses and swimming styles.

For a better generalisation, we propose, in this paper, an investigation of using a background removal technique for combining two datasets in order to further extend our existing dataset. In other words, we investigated the merging of components of two datasets involving humans as an additional augmentation technique. In the realm of background removal, an emerging trend is to incorporate deep learning-based approaches [28–30] to accomplish the background removal task. A comparison study in [29] revealed the practicability of YOLO models [20] in comparison to other feature engineering-based approaches [31,32] in removing distinct moving particles from their liquid background contexts. The proposed method in [28] incorporates an autoencoder along with a U-Net [33] to accomplish static and dynamic background generation tasks, respectively. Here, a set of static backgrounds were used to train an autoencoder. Later, the trained autoencoder was evaluated using a movement-free foreground, and the generated result was subtracted from the input image. This lead to a binary image that was used to train the U-Net, which requires pixel-level, labelled training images.

Our Contribution

Our main goal was to tackle the lack of available training images (“real images”) by employing and investigating a method of merging meaningful components of two datasets as an additional augmentation technique (“synthetic images”). Our investigations involved

coupling a deep learning-based background removal scheme called U2-Net [30] with YOLO models to comprise a robust vision-based system that can effectively operate in real-world environments. In general, we localised swimmers across a lake mainly using a YOLOv3 [18] architecture trained on a variety of real and synthetic images. We also provide results for a comparison with more modern versions like YOLOv5 and YOLOv8 [15]. Several combinations of the real and synthetic images were used to systematically form different datasets for investigating the accuracy of predicting swimming humans. Our datasets were additionally augmented to take into account various environmental conditions not covered by the very limited amount of training images we had. In total, we successfully created 17 training datasets (exploiting mixtures of real and synthetic imagery) and one completely different additional dataset for evaluation purposes (containing only real images) to investigate the impact of synthetically constructed images on prediction accuracy in realistic scenarios. Let us stress again that the motivation to produce the mentioned set of synthetic images is the lack of available datasets of outdoor swimmers related to our environment of operation. Our study represents a step forward in localising swimmers in a northern German outdoor lake environment under uncontrolled conditions.

2. YOLO Models

In the following, the YOLOv1 [16] model along with its advancements leading to the YOLOv3 architecture is discussed. Let us mention at this point that we used the YOLO framework as a tool, and the focus of this manuscript is on the training with various datasets. We included a brief discussion of YOLOv5 and YOLOv8 for the readers' convenience.

2.1. YOLOv1

The YOLOv1 architecture is visualised in Figure 1, and it was inspired by the GoogLeNet model [34] with 24 convolutional layers that downsamples the input training images of size $448 \times 448 \times 3$ to a $7 \times 7 \times 1024$ tensor (DarkNet Architecture in Figure 1). A grid of 7×7 cells was established, accounting for all of the three input image channels. YOLOv1 predicts, within each grid cell, two bounding boxes and their associated class probabilities. The $7 \times 7 \times 1024$ tensor is flattened into a $1 \times 1 \times 50,176$ vector for the fully connected input layer. After being processed through a $1 \times 1 \times 4096$ fully connected hidden layer, the fully connected output layer returns a vector of dimensions $1 \times 1 \times 1470$. In order to finally obtain a prediction for each of the 7×7 grid cells, the fully connected output vector is resized into a $7 \times 7 \times 30$ tensor, which is called a predictor tensor.

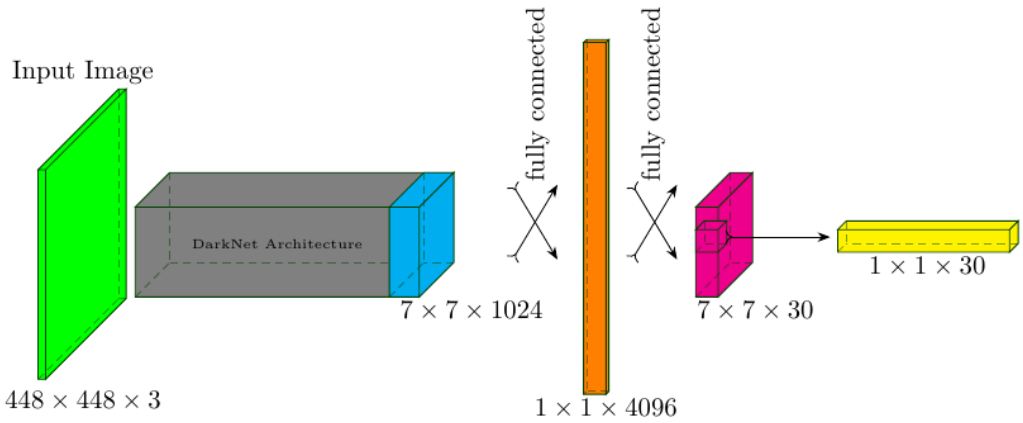


Figure 1. Illustration of the fundamental architecture behind the YOLOv1 framework.

Let us extract a $1 \times 1 \times 30$ predictor vector of the last layer, as shown in Figure 1 in yellow, and observe that it comprises 20 conditional class probabilities along with two vectors, each of size $1 \times 1 \times 5$. This conforms with the 20 labelled classes that appeared in the PASCAL Visual Object Classes Challenge (VOC) [35], while each of the $1 \times 1 \times 5$ predictors corresponds to two of the found bounding box properties, namely $\hat{x}$, $\hat{y}$, $\hat{h}$, and $\hat{w}$ as the centroid, height, width, and a confidence score [36]. The Intersection over Union (IoU) of the ground truth and the predicted bounding box is an important measurement metric in the YOLO model.

The YOLOv1 architecture merges the object detection and classification components into a unified framework [16]. This integration is achieved through a compound cost function (Equation (4)) consisting of three parts: *localisation loss* (Equation (1)), *confidence loss* (Equation (2)), and *classification loss* (Equation (3)). Together, these loss components contribute to the overall training objective of YOLOv1, enabling simultaneous object detection and classification tasks.

Let us start with the *localisation loss* (Equation (1)), which encourages the model to accurately predict the position coordinates and the height and width dimensions of the bounding boxes, namely $\hat{x}$, $\hat{y}$, $\hat{h}$, and $\hat{w}$. As the localisation loss is minimised, the model learns to adjust the predicted coordinates to align with the ground truth coordinates, leading to an improved localisation precision. With this, the localisation loss reads as follows:

$$\begin{aligned}\mathcal{L}_l(x, \hat{x}, y, \hat{y}, \lambda_{coord}) = \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] \\ + \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]\end{aligned} \tag{1}$$

where $\mathbb{1}_{ij}^{obj}$ is to be assigned the value 1, in case the j-th bounding box in the i-th cell is responsible for detecting an object, and 0 otherwise. Here, $B = 2$ is the number of bounding boxes predicted for each cell.

The confidence loss (Equation (2)) is applied to the object (ground truth) and the (predicted) class confidence score, C and $\hat{C}$, respectively, and penalises wrongly predicted objects enclosed with their bounding boxes across the grid cells of the input image:

$$\mathcal{L}_c\left(C_i, \hat{C}_i, \lambda_{noobj}\right) = \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} (C_i - \hat{C}_i)^2 + \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{noobj} (C_i - \hat{C}_i)^2 \tag{2}$$

where, in knowing that $\mathbb{1}_{ij}^{noobj}$ is a binary indicator variable, it takes the value 1 if the j-th bounding box in the i-th cell does not contain a detected object, and 0 otherwise. In this context, C and $\hat{C}$ are both in the range $[0, 1]$. The former is a multiplicative connection of the IoU and the probability of an object centroid being in the grid cell $Pr(object)$. This information comes from the ground truth and is either 1 or 0. Therefore, C is either equal to IoU or 0. $\hat{C}$ is learned during the training process, as it multiplicatively combines the IoU and the class probabilities $Pr(class_i, object)$ and $Pr(object)$. This results in a class-specific confidence score for each grid cell, since the class probability is only relevant when the centroid of an object is in the cell. In total, this score contains information about how good the class-box predictions in each cell are. The confidence score ground truth is not directly annotated by a human, but internally computed based on the ground truth bounding box.

The purpose of the coefficient λ_{coord} in localisation loss (Equation (1)) can be better understood when compared to the usage of λ_{noobj} appearing as part of the confidence loss in Equation (2). Both coefficients play a crucial role in mitigating model instability during the training phase. The λ_{noobj} penalises the loss associated with wrongly predicted objects in grid cells, namely, the background areas wrongly predicted as an object. A small value of λ_{noobj} reduces the number of false instances as the training proceeds. In

contrast, λ_{coord} encourages the importance of the accurate localisation of objects and their bounding boxes. The authors of YOLOv1 assigned a relatively smaller value of 0.5 to λ_{noobj} in Equation (2), while they considered a larger value of 5 for λ_{coord} in Equation (1). This choice effectively encourages the YOLOv1 model to focus on identifying bounding boxes that accurately represent objects while mitigating the impact of a falsely accepted background bounding boxes.

Finally, YOLOv1 incorporates the *classification loss* as follows:

$$\mathcal{L}_a\left(P_i(\kappa_s), \hat{P}_i(\kappa_s)\right) = \sum_{i=0}^{S^2} \mathbb{1}_i^{obj} \sum_{\kappa \in \text{classes}} \left(P_i(\kappa_s) - \hat{P}_i(\kappa_s)\right)^2 \tag{3}$$

where $P(\kappa_s)$ represents the ground truth class probability, namely, the class label, for class s in the j-th bounding box of the i-th grid cell. It is one-hot encoded, meaning that it is 1 for the targeted class and 0 for all other classes. The term $\mathbb{1}_i^{obj}$ is a binary indicator variable that takes the value 1 if the i-th cell contains an object, otherwise, 0. Note that $\hat{P}(\kappa_s)$ is estimated by the model during the training phase. To this end, the YOLOv1 total loss was deduced in [16] as

$$\mathcal{L} = \mathcal{L}_a + \mathcal{L}_c + \mathcal{L}_l. \tag{4}$$

2.2. YOLOv3

One notable advantage that sets YOLOv3 [18] apart from YOLOv1 is its ability to predict across multiple scales. This capability is accomplished through the incorporation of Darknet-53 [19], initially designed as a 53-layer network trained on ImageNet [37]. To enhance the detection capability, the number of layers in the network was doubled, leading to a comprehensive 106-layer fully convolutional architecture for YOLOv3. Within the stacked Darknet comprising 106 layers, upsampling and concatenation techniques are employed three times, generating feature maps with dimensions of $13 \times 13 \times 255$, $26 \times 26 \times 255$, and $52 \times 52 \times 255$. As documented in [18], the production of these feature maps involves upsampling the corresponding feature maps from the two preceding layers by a factor of 4. Subsequently, these upsampled maps are concatenated with their corresponding earlier feature maps from the network. According to the authors of [18], the utilisation of the technique referred to as the Feature Pyramid Network (FPN) [38] aims to acquire significant semantic information from the upsampled features and more detailed information from the earlier feature maps. Within YOLOv3, each output tensor with dimensions $1 \times 1 \times 255$ encompasses a total of $B = 3$ bounding boxes. These boxes are characterised by six attributes: centroid coordinates, dimensions, objectness score, and a set of 80 [39] conditional class confidences. With YOLOv3 making predictions on three distinct scales, a total of nine "derived" bounding boxes are anticipated. These derivations are executed using a set of predefined "anchor boxes", which are initially supplied to YOLOv3 during the preprocessing stage and are known as "dimension clusters" [17].

The primary purpose of anchor boxes is to establish a constrained set of predefined shapes derived from the dataset and the available ground truth boxes. This allows for a comparison during the training phase, where the ground truth boxes are matched against these anchors, and the model learns the transformations between the predefined anchors and the actual ground truth boxes. In this context, the model is trained by selecting the anchor box with the highest IoU with the ground truth box. In utilising a K-means clustering approach [40], a total of nine anchor boxes are determined, each representing the mean anchor box within one of nine established clusters across different scales. This clustering is performed in reference to the COCO dataset [39]. The primary advantage of these prior boxes lies in their ability to enhance the capacity of YOLOv3 to predict multiple objects, accommodating various height and width aspect ratios on different scales.

2.3. YOLOv5

In 2020, Ultralytics LLC introduced YOLOv5 [20,41]. Unlike Darknet-53 [19] in YOLOv3, a new model was built on PyTorch with a Cross Stage Partial Network (CSPNet) [42] as its backbone. This backbone enhances accuracy, reduces inference speed, and minimises model size by integrating gradient changes into feature maps. Utilising a *Path Aggregation Network* (PANet) [43] as a neck, YOLOv5 employs a *Feature Pyramid Network* (FPN) [38] to improve the low-level feature propagation, enhancing the object localisation accuracy. The YOLOv5 head resembles that of YOLOv3, generating three feature map outputs for multi-scale prediction. The image undergoes feature extraction in CSPDarknet-53, fusion in PANet, and final result generation in the YOLO layer. They also changed the classification loss function from cross-entropy in YOLOv3 to binary cross-entropy and a logits loss function in YOLOv5.

2.4. YOLOv8

Let us have a brief look at YOLOv8 [44–46]. It introduces several architectural updates, including a shift in the backbone from C3 to C2f, altering convolution sizes, and reconfiguring the bottleneck. Two convolutions in the YOLOv5 were removed. In YOLOv8, the bottleneck structure stays consistent with that of YOLOv5, except for one modification: the initial convolution kernel size has been enlarged from 1×1 to 3×3. This adjustment signifies a transition toward the ResNet [47] block introduced in 2015. YOLOv8 was designed for anchor-free detection, where the advantage of anchor-free detection is that it is more flexible and efficient, as it does not require the manual specification of anchor boxes. These can be difficult to choose, which may sometimes lead to suboptimal results in previous YOLO models such as in the v1 and v2. In YOLOv8, the loss function is divided into two main components—the classification branch employs binary cross-entropy (BCE) loss, and the regression branch, which handles bounding box prediction, incorporates a blend of two distinct losses: distribution focal loss (DFL) and complete Intersection over Union (CIoU) loss.

3. Data Preparation and Experimental Setup

In order to investigate the specific impact of real and synthetic training images on detection accuracy, we propose a systematic testing approach. We relate here the term *real images* to images of swimmers captured by a drone flying over a lake and *synthetic images* to images merged from two datasets as specified below. Let us stress the difficulty of our task that has roots in the dissimilarity of distributions from which the datasets are captured, namely, the lighting conditions, the distance of the camera to the swimmers, varying angles of view, and the limited availability of such data. At this point, we want to state that all images were captured with the consent of the visible people. The used images are protected by data privacy regulations.

3.1. Real Images

We extracted 150 real images from videos of a drone flying over a lake with one or more swimmers visible. The swimmers were captured in several different positions. Three example swimming styles are displayed in Figure 2. We used the original drone footage and cropped the images to a resolution of 416 by 416 for a more detailed representation of the swimming people.

Let us note at this point again that we target the issue of very limited data availability. That is, not only is the number of real images very limited, but also the number of swimming people, swimming styles, and poses. In addition, a major challenge is that the only available environmental condition in these images is sunshine. This causes light reflections on the water surface and may introduce issues for detection on, e.g., cloudy days. However, in the upcoming data augmentation paragraph we will further address this issue.

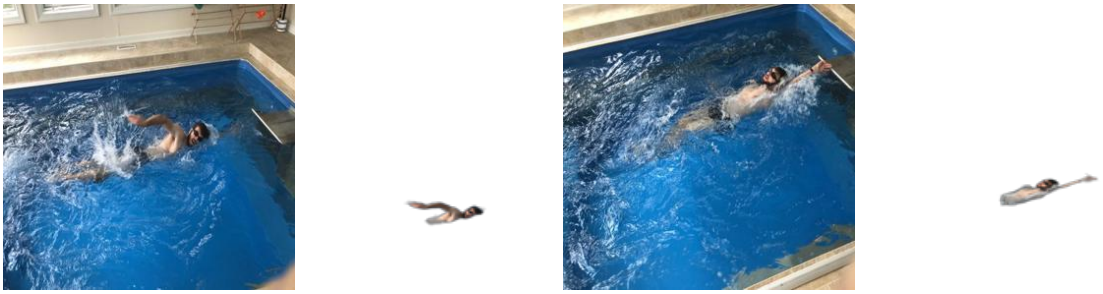
(**a**) *Real image* example 1 (**b**) *Real image* example 2 (**c**) *Real image* example 3

Figure 2. Three examples of one person, representing the *real images*. Different swimming styles may cause the appearance of foam around the swimmer. The limited number of swimmers and swimming styles available motivated the creation of *synthetic images*.

3.2. Synthetic Images

We also created 150 synthetic images based on the Kaggle dataset [48], which has 301 images, each of size 600 × 400 pixels. Those images were captured using a camera located at a relatively far distance from the objects of interest and its surrounding water. The Kaggle images were acquired in an indoor environment with predefined lighting conditions, so that the swimmer can be captured along with its background water.

One major contribution of our work is the proposed pipeline for merging two different distributions of datasets. Here, we consider some of the lake images that contain no swimmers as our target background. The motivation for this was to merge the lake water texture with the swimmers from the indoor environment. Our merging plan was to embed the indoor swimmers, with their backgrounds removed using U2-Net [30], in random positions and within the target backgrounds. Figure 3a–d show two swimmers from the Kaggle dataset and their background removed versions. Directly below, in Figure 3e–g, we display a subset of our merged images showing the lake background images with the extracted swimmers.



(**a**) Kaggle dataset example 1 (**b**) Background removed (**c**) Kaggle dataset example 2 (**d**) Background removed

(**e**) *Synthetic image* example 1 (**f**) *Synthetic image* example 2 (**g**) *Synthetic image* example 3

Figure 3. A pair of indoor pool swimmers (**a**,**c**) along with their corresponding background-removed images (**b**,**d**) obtained using U2-Net [30]. Swimmers of different swimming styles (**e**–**g**) already merged with lake backgrounds, representing the *synthetic images*.

3.3. Overview of the Image Datasets as Used in Our Work

Having explained above how we obtained real and synthetic images, we now turn to their usage. For the construction of the underlying *training dataset*, we will explore several different compositions of augmented real and synthetic images, as will be explained subsequently in detail in the experimental section. For training YOLOv3, it is standard to partition the training dataset, taking (in a random way) 80% for training the network, 10% for validation, i.e., for tuning hyperparameters and the optimisation, and again 10% for testing [49]. We also did this in our work, obtaining, in this way, a *validation dataset* and a *test dataset*. These are subsets of the underlying training dataset.

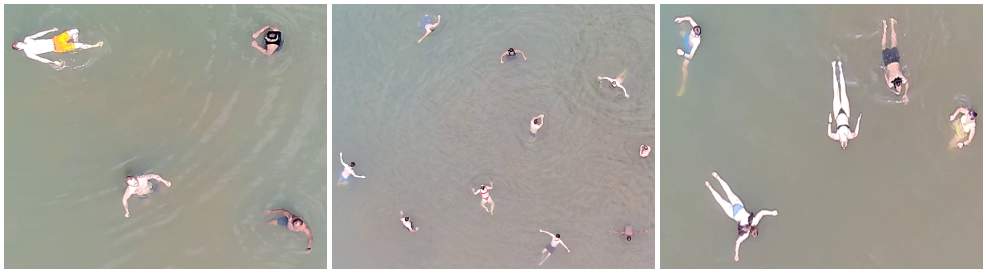
Let us explicitly point out that the test dataset contains, in general, not only real world imagery, but also synthetic images, since our aim was to explore the effect of the compositions of real and synthetic images in the training dataset. Therefore, it appears not to be useful to employ the test dataset for the actual quantitative evaluation of our approach, as the evaluation should bear a meaningful implication in a real-world setting, where no synthetic images are provided.

Consequently, for the actual evaluation, we mainly considered a separate dataset that we will call the *evaluation dataset*, which was constructed as explained below and that contains only real-world images.

3.4. Images Used for Quantitative Assessment: Evaluation Dataset

In order to have a meaningful evaluation of the results, we mainly used a completely separate and different dataset to evaluate the results. To make this point explicit, this separate dataset was not contained in the training dataset, so there is also no relation to the test dataset.

The images were also taken by a drone flying over a lake but, on a different day (which induced somewhat different lighting) and with different people. In contrast to the real images from the training dataset, up to ten persons are visible in the evaluation set, an account of which can be seen in Figure 4. We carefully selected these images for the evaluation dataset by taking into account substantial differences compared to the images in the training dataset, namely, more people being visible in one image, either standing in the water or showing additional swimming styles on a different background. In total, the evaluation dataset consists of 50 images.



(**a**) *Evaluation image* example 1 (**b**) *Evaluation image* example 2 (**c**) *Evaluation image* example 3

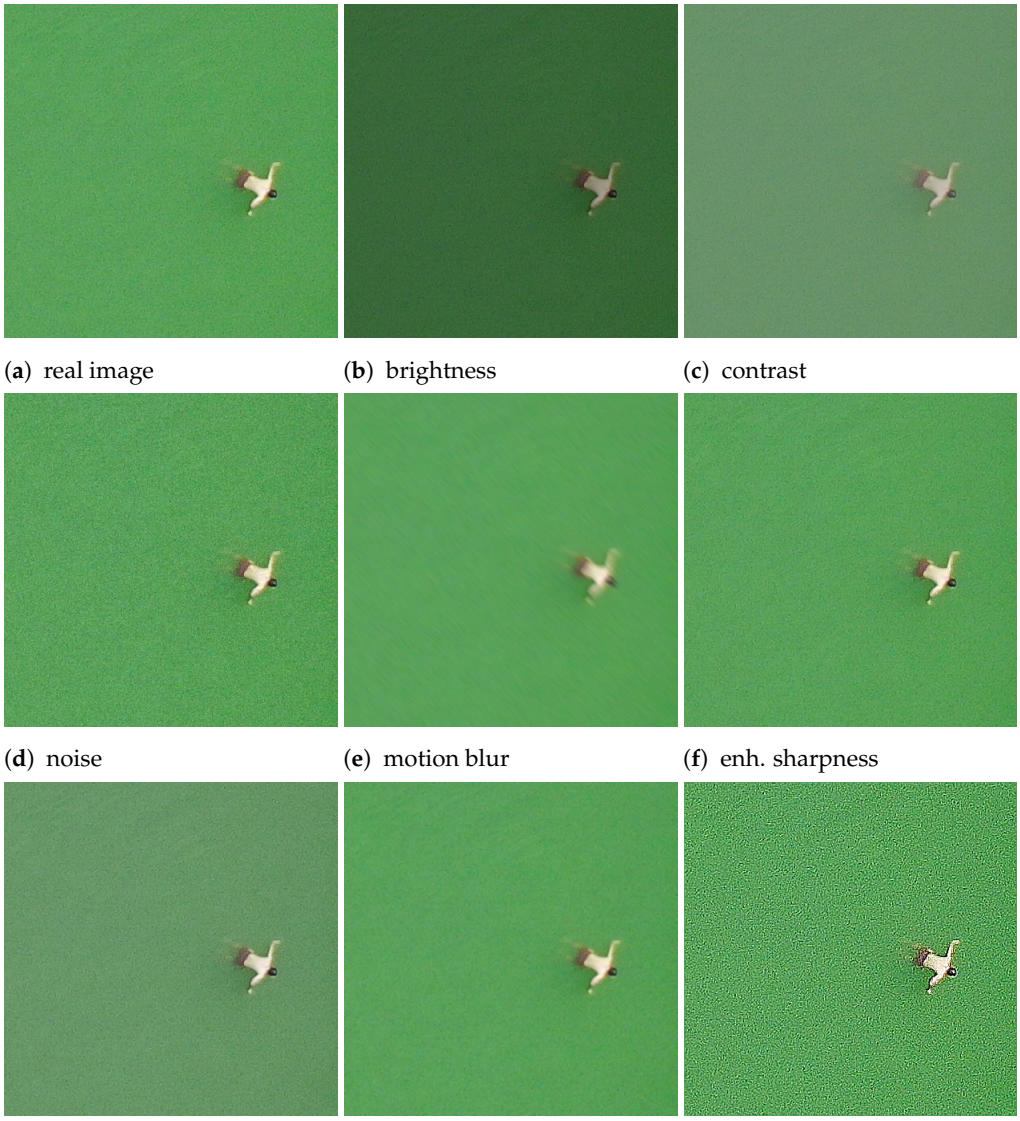
Figure 4. Representations of three examples from the *evaluation dataset*, which are different from the real images in the *training dataset*. Differences between these datasets are, e.g., the number of visible persons showing various swimming styles, the background (colour and texture), and camera settings, which e.g., translates to different apparent sizes of swimmers.

3.5. Data Augmentation

We augmented the very limited number of training images to create the datasets for our investigations. That is, editing images to change certain characteristics can simulate environmental conditions and increase the available number of images for training.

More precisely, we applied image processing techniques to change the following characteristics: *(i)* brightness (simulate different lighting conditions), *(ii)* contrast (enhance differences between lighter and darker regions), *(iii)* noise (simulate challenging lighting

conditions, different distances and weather conditions), *(iv)* motion blur (simulate camera movement blur), *(v)* enhance sharpness (enhance differences between objects and lighting condition changes), *(vi)* enhance colour (make objects look different), *(vii)* smooth (enhance the merging of synthetic images), *(viii)* enhance edge (increase texture visibility). Figure 5b–i show examples of the augmentations applied to a real image (Figure 5a). Although some augmentation techniques have similar results, the features extracted using CNN-based frameworks like YOLOv3 are always different and add flexibility to the dataset. The image augmentation techniques were implemented using the library imgaug [50].



(**a**) real image (**b**) brightness (**c**) contrast

(**d**) noise (**e**) motion blur (**f**) enh. sharpness

(**g**) enh. colour (**h**) smooth (**i**) enh. edge

Figure 5. Visualisation of the utilised augmentation techniques for an example image. Each image (real and synthetic) was augmented several times with each method and different parameters; "enh." is short for "enhance".

3.6. Experimental Setup

We prepared 17 training datasets in total that were used in two experiments to investigate the following:

1. The impact of replacing real images with synthetic images.
2. The benefits of adding synthetic images to real images.

In Experiment 1, we trained YOLOv3 eleven times, where each of the corresponding eleven training datasets had a fixed amount of 150 (raw) images before augmentation. The images were augmented 40 times each, so that in total, one training dataset consisted of 6150 images. The 150 raw images were entirely covered by real images in the first of these datasets. Subsequently, the real images were gradually replaced by the synthetic images, as shown in Table 1.

Table 1. Training dataset composition of real and synthetic images for each setup in Experiment 1 and Experiment 2.

Dataset	Experiment 1	Experiment 2
1	150 real + 0 synthetic	150 real + 25 synthetic
2	135 real + 15 synthetic	150 real + 50 synthetic
3	120 real + 30 synthetic	150 real + 75 synthetic
4	105 real + 45 synthetic	150 real + 100 synthetic
5	90 real + 60 synthetic	150 real + 125 synthetic
6	75 real + 75 synthetic	150 real + 150 synthetic
7	60 real + 90 synthetic	
8	45 real + 105 synthetic	
9	30 real + 120 synthetic	
10	15 real + 135 synthetic	
11	0 real + 150 synthetic	

For Experiment 2, we trained YOLOv3 six times with a continuously increasing number of raw images, achieved by adding more and more synthetic images to the 150 real images, as displayed in Table 1. Again, each image was augmented 40 times using the above-mentioned image processing techniques. Let us note explicitly that in this way, we obtained six different training datasets with different numbers of images.

4. Results and Discussion

Let us highlight again the importance of detection accuracy when it comes to swimmer safety in order to ensure the correct recognition of people struggling in water environments. Testing the approach of background removal and merging two datasets as an additional augmentation technique were conducted to enhance the robustness of swimmer localisation in northern German lake environments. This approach was combined with YOLO real-time object detection frameworks.

To quantify the results, we considered the mean average precision (mAP) metric. More specifically, both mAP@.5 and mAP@.5:.95. The metrics use the average precision measure that results from the Intersection over Union (IoU) of the ground truth bounding box and the predicted bounding box. Defining a threshold for the IoU determines if a detection is a True Positive (TP), when the IoU is above the threshold, or False Positive (FP), otherwise. The letter means, in other words, that the model has made a prediction that does not overlap enough with the ground truth. A False Negative (FN) detection occurs when the model predicts a correct instance as false. The precision is then computed as the number of TPs over the sum of TPs and FPs. One also needs to consider the recall, which is computed as the TPs over the sum of TPs and FNs. The precision is then plotted over the recall, and the area below the graph defines the average precision (AP). An additional averaging of the APs for all classes (e.g., dog, cat) defines the mAP. Setting the IoU threshold to 0.5 (or 50%) returns the mAP@.5 metric. More meaningful, however, is mAP@.5:.95. Here, we have several thresholds, iteratively ranging from 0.5 to 0.95 in steps of 0.05. The mAP is then averaged (again) over all thresholds, resulting in mAP@.5:.95.

For training, we utilised large YOLOv3, YOLOv5, and YOLOv8 architectures in the first part together with 100 epochs. As indicated already, the *training datasets* were separated for training (80%), testing (10%), and validation (10%), as it is required for the YOLO frameworks. For evaluation, as discussed, we used (if not mentioned otherwise) a separate *evaluation dataset* with 50 real images, also taken by a drone, showing different

persons and conditions (captured on another day); see again Figure 4. For meaningful comparisons and discussions, we kept all changeable parameters (e.g., batch size, epochs, augmentation techniques) constant. The only variable is the training dataset composition (see Table 1). This allowed us to focus on the latter in combination with real-time object detection without taking other factors into account.

4.1. YOLOv3 vs. YOLOv5 vs. YOLOv8

Prior to the actual experiments ran with YOLOv3, we found results comparing YOLOv3, YOLOv5, and YOLOv8, as shown in Table 2. The mAP was derived here from an evaluation with the completely unseen evaluation dataset, containing only real images.

Table 2. Evaluation mAP comparison for three different YOLO versions in the context of our training datasets. For example, the notation "exp1-6" refers to Experiment 1, dataset 6 (see Table 1). Let us note again that the mAPs are computed with respect to the evaluation dataset.

Dataset	YOLOv3		YOLOv5		YOLOv8	
	mAP@.5	mAP@.5:.95	mAP@.5	mAP@.5:.95	mAP@.5	mAP@.5:.95
exp1-1	0.960	0.770	0.935	0.725	0.979	0.780
exp1-6	0.983	0.797	0.963	0.812	0.985	0.825
exp1-11	0.911	0.727	0.904	0.656	0.245	0.183
exp2-1	0.983	0.797	0.666	0.520	0.949	0.751
exp2-3	0.941	0.764	0.719	0.489	0.995	0.794
exp2-6	0.967	0.814	0.625	0.476	0.931	0.720

There are indications that for investigating our datasets, YOLOv3 and YOLOv8 work best with a remarkable mAP@.5 = 0.995 for exp2-3 and YOLOv8. However, especially with the increased presence of synthetic images, YOLOv3 outperforms the other versions, which strengthens our ambitions to work with YOLOv3 in the subsequent sections. We also want to highlight the consistency of YOLOv3, where each mAP@.5 is above 0.911 and each mAP@.5:.95 is above 0.727, in contrast to the other versions.

We find, in Table 3, the mAP results for YOLOv3 training five times. Except for different batch sizes, we kept all other settings and the test dataset, as well as the evaluation dataset, equal. The training with batch size 16 (emphasised in the table) led to the results in accordance with the findings in Table 2. This provides insight into the performance (in the context of our datasets) and supports the use of YOLOv3 in the subsequent investigations because of the consistently high precision.

Table 3. mAP results based on training YOLOv3 five times in Experiment 1, on dataset 6 (see Table 1, exp1-6), using different batch sizes. Let us note that the comparison here is with respect to both the test dataset and the evaluation dataset.

Number	Batch Size	Test Dataset		Evaluation Dataset	
		mAP@.5	mAP@.5:.95	mAP@.5	mAP@.5:.95
1	128	0.995	0.995	0.907	0.682
2	64	0.995	0.994	0.928	0.713
3	32	0.995	0.994	0.924	0.697
4	16	0.995	0.995	0.983	0.797
5	8	0.995	0.993	0.944	0.731

In both tables we find the exact same results for batch size 16, although YOLOv3 was trained individually. The reason for this behaviour is that we trained our custom dataset on a pretrained YOLO model, which is the common approach. Therefore, no initial randomness was introduced (like, e.g., random initial weights would), and training without changing parameters or settings returns an equal result.

4.2. Experiment 1: Replacing Real Images with Synthetic Images

Let us turn to Experiment 1, where we investigated the impact of replacing real images with synthetic images in the training dataset (see Table 1).

In Figure 6 we find the mAP@.5 and mAP@.5:.95 plotted over the datasets. Please see Table 1 for the specific combinations of real and synthetic images per training dataset.

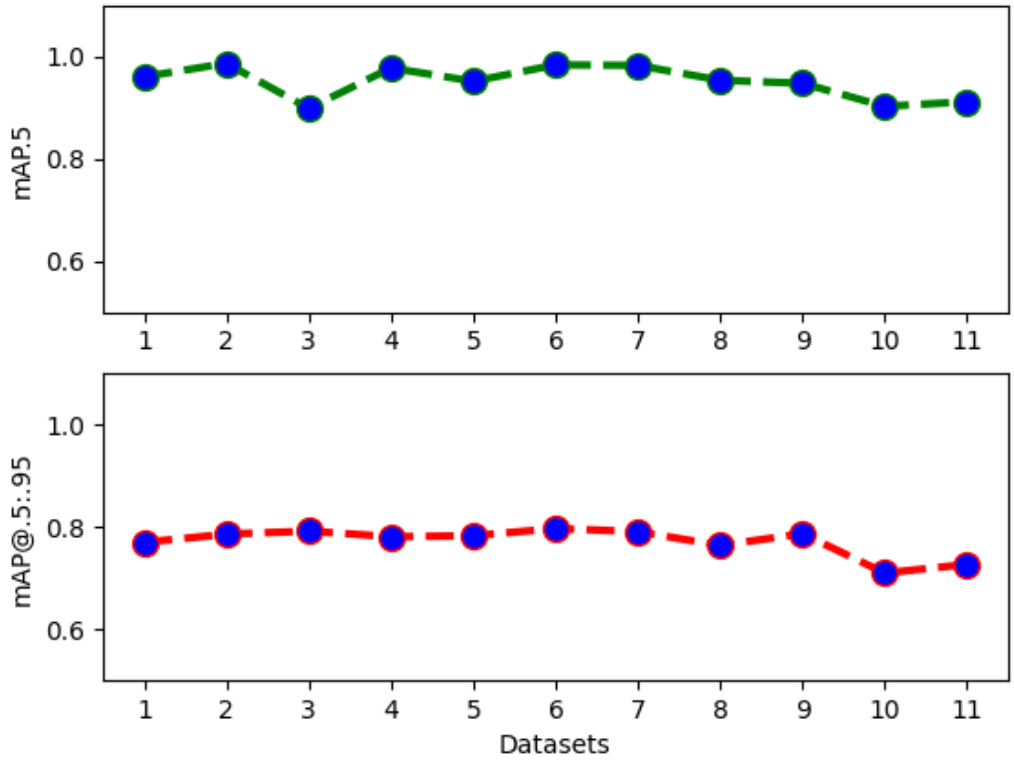


Figure 6. Results for both mAP@.5 and mAP@.5:.95 metrics for Experiment 1.

At a qualitative level, there is no general trend visible. When it comes to a quantitative analysis, dataset 2 (90% real, 10% synthetic) provides the best mAP@.5 = 0.986, followed by dataset 6 (50% real, 50% synthetic) with mAP@.5 = 0.983 and dataset 7 (40% real, 60% synthetic) with mAP@.5 = 0.982. In the ranking for the second metric, dataset 6 has the best mAP@.5:.95 = 0.797, followed by dataset 3 (80% real, 20% synthetic) with mAP@.5:.95 = 0.792 and dataset 7 with mAP@.5:.95 = 0.791. While the worst results were always associated with datasets 10 and 11, dataset 1, with 100% real images, was somewhere in the middle, performance-wise.

These results are clear indications that features extracted from the synthetic images have similar characteristics to those of the real swimmers. One reason is related to the fact that the swimming persons stand out from the background, just like the merged swimmers in the synthetic images. However, since the evaluation dataset is completely different from the training dataset, especially with more swimming styles and poses, a different background, and small waves, we find these results to be meaningful on different levels.

In Figure 7, we can see one image from the evaluation dataset, detected using the trained YOLOv3 framework for dataset 1 (100% real, 0% synthetic), dataset 6 (50% real, 50% synthetic), and dataset 11 (0% real, 100% synthetic). These detections confirm the results from Figure 6, where dataset 6 shows one of the best results. While in Figure 7a, the swimmer on the left was detected with a prediction of 0.56, the detection of this person failed in Figure 7c. On the other hand, this swimmer was predicted with 0.98 for the dataset using 50% real and 50% synthetic images.

(**a**) Detection based on dataset 1 (**b**) Detection based on dataset 6 (**c**) Detection based on dataset 11

Figure 7. Detections based on the results from YOLOv3 training in Experiment 1.

Conclusions on Experiment 1

To conclude the first experiment, we saw evidence that mixing real images with constructed (merged/synthetic) images can be beneficial for the detection accuracy. However, a clear statement of which ratio is the best is difficult to make. The best results are associated with the following splits of images: 90% real + 10% synthetic, 50% real + 50% synthetic and 40% real + 60% synthetic.

4.3. Experiment 2: Adding Synthetic Images to Real Images

In Experiment 2, we investigated the impact of adding synthetic images to the real images in order to see whether this is beneficial or not.

In order to evaluate the detection accuracy of YOLOv3 based on the training datasets from Table 1, we again considered the mAP@.5 and mAP@.5:.95, as shown in Figure 8. The quantitative results for dataset 1 (150 real + 25 synthetic) returned mAP@.5 = 0.983, which was the highest in this category and still better than that of dataset 1 from the previous experiment (mAP@.5 = 0.960) with only 150 real images and no synthetic images. We again found indications that the constructed images have a beneficial effect and datasets can have a certain amount of synthetic images to improve the accuracy. On the other hand, the results for mAP@.5:.95 had an interesting behaviour. Dataset 6 (150 real + 150 synthetic) had the highest mAP@.5:95 = 0.814 of all the datasets, including every single dataset from the previous experiment. This is, in general, a desirable result, as we find it to indicate a more careful selection of the detected (less FPs) objects, while perhaps the detection itself is in some cases a little bit less good.

The predictions for dataset 1 (150 real + 25 synthetic), dataset 3 (150 real + 75 synthetic), and dataset 6 (150 real + 150 synthetic) are exemplarily shown in Figure 9. In comparison to Figure 7, the current evaluation image has slight changes and is in fact another frame from the same sequence. However, even minor changes may have an impact on the detection, as the water texture (like waves and foam) and swimming styles change. We clearly observe that the prediction accuracy throughout the three images increases. However, the person at the top was detected as two different swimmers, as shown in Figure 9a,c.

In the previous paragraph, we stated that an increasing mAP@.5:.95 indicates a more careful selection of the detected objects. This statement continues to hold, since the double detection of the same swimmer is still a TP prediction, even with a better accuracy. The main difference between the detections in Figure 9 is the swimming person on the left, who was only directly detected with dataset 6, and with high accuracy. We find this to be a positive effect of the added synthetic images. However, adding more (unaugmented) images to a dataset, which represent different situations, should, in theory, always have a beneficial impact. Nonetheless, this behaviour clearly indicates that the synthetic images serve their intended purpose.

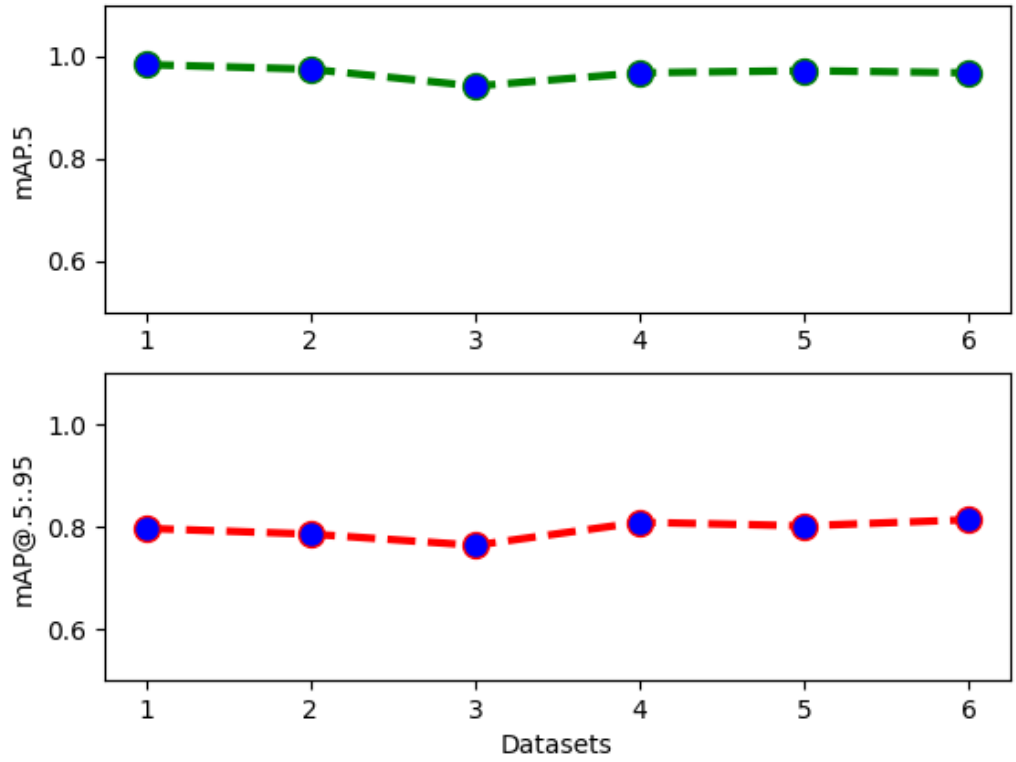
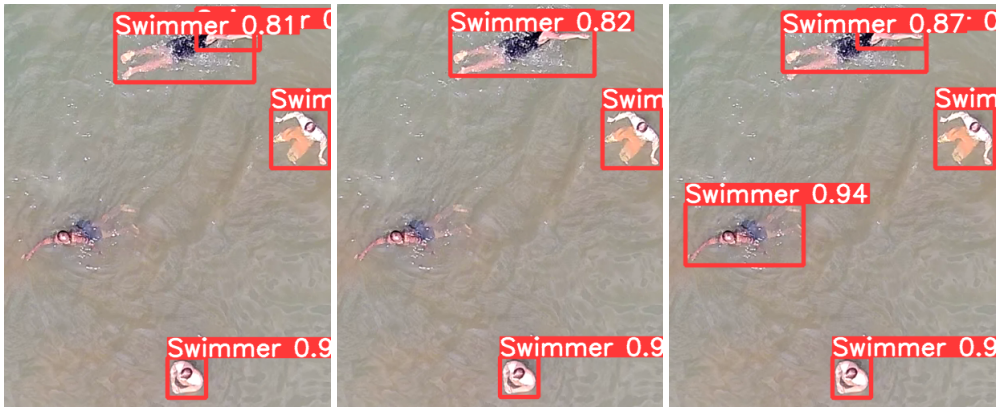


Figure 8. Results for both mAP@.5 and mAP@.5:.95 metrics for Experiment 2.



(**a**) Detection based on dataset 1 (**b**) Detection based on dataset 3 (**c**) Detection based on dataset 6
Figure 9. Detections based on the results from the YOLOv3 training in Experiment 2.

Conclusions

Concluding this experiment, we found that adding synthetic images to a fixed amount of real images has a benefit. An equal amount of both real and synthetic images returned the best predictions. Therefore, we found that the constructed synthetic images are good representations of real swimming persons.

4.4. Detection on Real Images with Objects

Finally, we wanted to share some results for the detection of swimming humans with objects in the same image, which can be seen in Figure 10.

Figure 10. Real images that include ball, grasses, sun reflections, and also waves as barriers to show the robustness of the YOLO settings explored in Experiment 2, dataset 1 (see Table 1).

Even with the very limited variety of unaugmented images (150 real + 25 synthetic) in our training dataset, the model can identify swimming persons in these real-world evaluation images. On top of that, the additional visible objects like, e.g., grass, balls, and persons outside the water were correctly not detected. Let us note again at this point that the focus of the presented work was to study whether creating synthetic images for swimmer detection can be utilised with benefit as an additional augmentation. The overall results support this assumption. However, common objects in lake and beach environments like, e.g., balls, grass, and air mattresses, as well as non-swimming humans inside or outside the water, should also be part of the detection to avoid possible prediction errors and make the framework more robust.

5. Discussion and Future Work

In the current study, we studied an approach to localising and classifying swimmers across a lake captured by a drone. Swimmer safety in outdoor lake environments, such as here in northern Germany, is a topic of great importance and so is the responsibility for a properly working approach when people rely on the system.

Our approach uses a large YOLOv3 model with focus on different training datasets consisting of real and synthetic images. The latter were constructed/merged from two different datasets. The datasets were systematically setup, augmented, and investigated regarding their impact on the mAP accuracy measure. The trained YOLOv3 networks were tested on a completely different evaluation dataset.

We saw that replacing real images with synthetic images for a fixed number of unaugmented images in a dataset has its benefits. The ratio showing the best results was 50%. We can also say that adding synthetic images to a fixed number of real images has a positive effect on the robustness of the detection. There are clear indications that continuing research on constructing synthetic images for datasets with a very limited number of real images is necessary since utilising this approach as an additional augmentation technique to enhance swimmer safety in an outdoor lake environment seems to be beneficial.

Future work will focus on the next steps toward a robust swimmer safety pipeline that helps in saving people who struggle in lake environments based on, e.g., muscle cramps, heart attacks, or a lack of swimming skills. Improving the appearance of synthetic images will be a major component, including different compositions with the testing of additional background removal techniques. Background images will also be merged with objects like, e.g., boats, air mattresses, and vegetation. The collection of additional images in the context of our dataset is planned to be initiated.

Author Contributions: The authors declare that the main idea and experiments were proposed by M.K.M. and that the manuscript was written together with T.S., A.M.Y. and M.B. All authors have read and agreed to the published version of the manuscript.

Funding: The authors acknowledge funding of their work by Bundesministerium für Digitales und Verkehr (BMDV) within the project *RescueFly* as well as by Bundesministerium für Bildung und Forschung (BMBF) within the project *KI@MINT* ("AI-Lab").

Institutional Review Board Statement: The study was approved by the Ethics Committee of the Brandenburg University of Technology Cottbus-Senftenberg (protocol code: EK2024-07, date of approval: 22 April 2024).

Informed Consent Statement: Informed consent was obtained from all subjects involved in the study, according to the data privacy regulations (DSGVO) of the European Union.

Data Availability Statement: The authors declare that upon reasonable request, the data and the code are available from the corresponding author.

Acknowledgments: The authors acknowledge all people who helped in capturing the lake images for the dataset. This especially includes the organisers, the visible people, and the drone-flying team. We would also like to thank the unknown reviewers for their constructive and valuable comments that led to a substantial improvement of our paper.

Conflicts of Interest: The authors have no conflicts of interest to declare that are relevant to the content of this article.

Abbreviations

The following abbreviations are used in this manuscript:

FPN	Feature Pyramid Network;
IoU	Intersection over Union;
mAP	mean Average Precision;
TP	True Positive;
FP	False Positive;
FN	False Negative;
CNN	Convolutional Neural Network;
YOLO	You Only Look Once.

References

1. Shatnawi, M.; Albreiki, F.; Alkhoori, A.; Alhebshi, M. Deep Learning and Vision-Based Early Drowning Detection. *Information* **2023**, *14*, 52. [CrossRef]
2. Xiao, H.; Li, Y.; Xiu, Y.; Xia, Q. Development of outdoor swimmers detection system with small object detection method based on deep learning. *Multimed. Syst.* **2022**, *29*, 323–332. [CrossRef]
3. Cafarelli, D.; Ciampi, L.; Vadicamo, L.; Gennaro, C.; Berton, A.; Paterni, M.; Benvenuti, C.; Passera, M.; Falchi, F. MOBDrone: A Drone Video Dataset for Man OverBoard Rescue. In Proceedings of the Image Analysis and Processing—ICIAP 2022, Lecce, Italy, 23–27 May 2022; Springer: Cham, Switzerland, 2022; pp. 633–644.
4. Handalage, U.; Nikapotha, N.; Subasinghe, C.; Prasanga, T.; Thilakarthna, T.; Kasthurirathna, D. Computer Vision Enabled Drowning Detection System. In Proceedings of the 2021 3rd International Conference on Advancements in Computing (ICAC), Colombo, Sri Lanka, 9–11 December 2021; pp. 240–245. [CrossRef]
5. "Drowning", 25 July 2023. Available online: https://www.who.int/news-room/fact-sheets/detail/drowning (accessed on 12 March 2024).
6. *Drowning—United States, 2005–2009*; CDC: Atlanta, GA, USA, 2012.
7. Seguin, C.; Blaquière, G.; Loundou, A.; Michelet, P.; Markarian, T. Unmanned aerial vehicles (drones) to prevent drowning. *Resuscitation* **2018**, *127*, 63–67. [CrossRef] [PubMed]
8. Piccardi, M. Background subtraction techniques: A review. In Proceedings of the 2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No. 04CH37583), The Hague, The Netherlands, 10–13 October 2004; Volume 4, pp. 3099–3104. [CrossRef]
9. Georgakis, G.; Mousavian, A.; Berg, A.C.; Kosecka, J. Synthesizing training data for object detection in indoor scenes. *arXiv* **2017**, arXiv:1702.07836.
10. Benarab, D.; Napoléon, T.; Alfalou, A.; Verney, A.; Hellard, P. Swimmer's Head Detection Based on a Contrario and Scaled Composite JTC Approaches. *Int. J. Opt.* **2020**, *2020*, 4145938. [CrossRef]

11. Pogalin, E.; Thean, A.H.C.; Baan, J.; Schipper, N.W.; Smeulders, A.W.M. Video-based training registration for swimmers. *Int. J. Comput. Sci. Sport* **2007**, *6*, 4–17.
12. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. *arXiv* **2013**, arXiv:1311.2524 .
13. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask R-CNN. *arXiv* **2017**, arXiv:1703.06870.
14. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *arXiv* **2015**, arXiv:1506.01497.
15. Terven, J.; Cordova-Esparza, D. A comprehensive review of YOLO: From YOLOv1 to YOLOv8 and beyond. *arXiv* **2023**, arXiv:2304.00501.
16. Redmon, J.; Divvala, S.K.; Girshick, R.B.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. *arXiv* **2015**, arXiv:1506.02640.
17. Redmon, J.; Farhadi, A. YOLO9000: Better, Faster, Stronger. *arXiv* **2016**, arXiv:1612.08242.
18. Redmon, J.; Farhadi, A. YOLOv3: An Incremental Improvement. *arXiv* **2018**, arXiv:1804.02767.
19. Bochkovskiy, A.; Wang, C.; Liao, H.M. YOLOv4: Optimal Speed and Accuracy of Object Detection. *arXiv* **2020**, arXiv:2004.10934.
20. Jocher, G.; Chaurasia, A.; Stoken, A.; Borovec, J.; NanoCode012; Kwon, Y.; Tao, X.; Michael, K.; Fang, J.; Imyhxy; et al. ultralytics/yolov5: v6.2—YOLOv5 Classification Models, Apple M1, Reproducibility, ClearML and Deci.ai Integrations (v6.2). Zenodo 2022. Available online: https://zenodo.org/records/7002879 (accessed on 12 March 2024).
21. Divvala, S.K.; Hoiem, D.; Hays, J.H.; Efros, A.A.; Hebert, M. An empirical study of context in object detection. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA , 20–25 June 2009; pp. 1271–1278. [CrossRef]
22. Ren, S.; He, K.; Girshick, R.; Zhang, X.; Sun, J. Object Detection Networks on Convolutional Feature Maps. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1476–1481. [CrossRef] [PubMed]
23. Chlap, P.; Min, H.; Vandenberg, N.; Dowling, J.; Holloway, L.; Haworth, A. A review of medical image data augmentation techniques for deep learning applications. *J. Med. Imaging Radiat. Oncol.* **2021**, *65*, 545–563. [CrossRef] [PubMed]
24. Zoph, B.; Cubuk, E.D.; Ghiasi, G.; Lin, T.Y.; Shlens, J.; Le, Q.V. Learning Data Augmentation Strategies for Object Detection. ECCV 2020. In *Lecture Notes in Computer Science*; Springer: Cham, Switzerland, 2020; Volume 12372, pp. 566–583. [CrossRef]
25. Shorten, C.; Khoshgoftaar, T.M. A survey on image data augmentation for deep learning. *J. Big Data* **2019**, *6*, 60. [CrossRef]
26. Yarahmadi, A.M.; Breuß, M.; Mohammadi, M.K. Explaining StyleGAN Synthesized Swimmer Images in Low-Dimensional Space. In Proceedings of the Computer Analysis of Images and Patterns, Limassol, Cyprus, 25–28 September 2023; Springer: Cham, Switzerland, 2023; pp. 164–173.
27. Sha, L.; Lucey, P.; Morgan, S.; Pease, D.L.; Sridharan, S. Swimmer Localization from a Moving Camera. In Proceedings of the 2013 International Conference on Digital Image Computing: Techniques and Applications (DICTA), Hobart, Australia, 26–28 November 2013; pp. 1–8.
28. Bahri, F.; Ray, N. Weakly Supervised Realtime Dynamic Background Subtraction. *arXiv* **2023**, arXiv:2303.02857.
29. Kara, E.; Zhang, G.; Williams, J.J.; Ferrandez-Quinto, G.; Rhoden, L.J.; Kim, M.; Kutz, J.N.; Rahman, A. Deep Learning Based Object Tracking in Walking Droplet and Granular Intruder Experiments. *arXiv* **2023**, arXiv:2302.05425.
30. Qin, X.; Zhang, Z.; Huang, C.; Dehghan, M.; Zaiane, O.R.; Jagersand, M. U2-Net: Going deeper with nested U-structure for salient object detection. *Pattern Recognit.* **2020**, *106*, 107404. [CrossRef]
31. Zivkovic, Z. Improved adaptive Gaussian mixture model for background subtraction. In Proceedings of the 17th International Conference on Pattern Recognition, ICPR 2004, Cambridge, UK, 23–26 August 2004; Volume 2, pp. 28–31. [CrossRef]
32. Zivkovic, Z.; van der Heijden, F. Efficient adaptive density estimation per image pixel for the task of background subtraction. *Pattern Recognit. Lett.* **2006**, *27*, 773–780. [CrossRef]
33. Ronneberger, O.; Fischer, P.; Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. In Proceedings of the Medical Image Computing and Computer-Assisted Intervention—MICCAI 2015, Munich, Germany, 5–9 October 2015; Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F., Eds.; Springer: Cham, Switzerland, 2015; pp. 234–241.
34. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going Deeper with Convolutions. *arXiv* **2014**, arXiv:1409.4842. [CrossRef]
35. Everingham, M.; Eslami, S.M.; Gool, L.; Williams, C.K.; Winn, J.; Zisserman, A. The Pascal Visual Object Classes Challenge: A Retrospective. *Int. J. Comput. Vision* **2015**, *111*, 98–136. [CrossRef]
36. Rezatofighi, H.; Tsoi, N.; Gwak, J.; Sadeghian, A.; Reid, I.; Savarese, S. Generalized Intersection over Union: A Metric and a Loss for Bounding Box Regression. *arXiv* **2019**, arXiv:1902.09630. [CrossRef]
37. Deng, J.; Dong, W.; Socher, R.; Li, L.J.; Li, K.; Li, F.-F. Imagenet: A large-scale hierarchical image database. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; IEEE: Piscataway, NJ, USA, 2009; pp. 248–255.
38. Lin, T.Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature Pyramid Networks for Object Detection. *arXiv* **2016**, arXiv:1612.03144. [CrossRef]
39. Lin, T.; Maire, M.; Belongie, S.J.; Bourdev, L.D.; Girshick, R.B.; Hays, J.; Perona, P.; Ramanan, D.; Zitnick, C.L.; Dollár, P. Microsoft COCO: Common Objects in Context. *arXiv* **2014**, arXiv:1405.0312.
40. Lloyd, S. Least squares quantization in PCM. *IEEE Trans. Inf. Theory* **1982**, *28*, 129–137. [CrossRef]

41. Jocher, G.; Chaurasia, A.; Stoken, A.; Borovec, J.; NanoCode012; Kwon, Y.; Michael, K.; Tao, X.; Fang, J.; Imyhxy; et al. ultralytics/yolov5: v7.0—YOLOv5 SOTA Realtime Instance Segmentation (v7.0). Zenodo 2022. Available online: https://ieeexplore.ieee.org/document/5206532 (accessed on 12 March 2024).
42. Wang, C.Y.; Liao, H.Y.M.; Yeh, I.H.; Wu, Y.H.; Chen, P.Y.; Hsieh, J.W. CSPNet: A New Backbone that can Enhance Learning Capability of CNN. *arXiv* **2019**, arXiv:1911.11929.
43. Liu, S.; Qi, L.; Qin, H.; Shi, J.; Jia, J. Path Aggregation Network for Instance Segmentation. *arXiv* **2018**, arXiv:1803.01534. [CrossRef]
44. Terven, J.; Córdova-Esparza, D.M.; Romero-González, J.A. A Comprehensive Review of YOLO Architectures in Computer Vision: From YOLOv1 to YOLOv8 and YOLO-NAS. *Mach. Learn. Knowl. Extr.* **2023**, *5*, 1680–1716. [CrossRef]
45. Talaat, F.M.; ZainEldin, H. An improved fire detection approach based on YOLO-v8 for smart cities. *Neural Comput. Appl.* **2023**, *35*, 20939–20954. [CrossRef]
46. Jocher, G.; Chaurasia, A.; Qiu, J. Ultralytics YOLO (Version 8.0.0) [Computer Software]. 2023. Available online: https://github.com/ultralytics/ultralytics (accessed on 12 March 2024).
47. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. *arXiv* **2015**, arXiv:1512.03385.
48. Coughlin, S. Swimmers. 2021. Available online: https://www.kaggle.com/datasets/seanmc4/swimmers (accessed on 12 March 2024).
49. Xu, Y.; Goodacre, R. On Splitting Training and Validation Set: A Comparative Study of Cross-Validation, Bootstrap and Systematic Sampling for Estimating the Generalization Performance of Supervised Learning. *J. Anal. Test.* **2018**, *2*, 249–262. [CrossRef] [PubMed]
50. Jung, A.B.; Wada, K.; Crall, J.; Tanaka, S.; Graving, J.; Reinders, C.; Yadav, S.; Banerjee, J.; Vecsei, G.; Kraft, A.; et al. Imgaug [Computer Software]. 2020. Available online: https://github.com/aleju/imgaug (accessed on 12 March 2024).