Article

Not peer-reviewed version

# Object Detection in Adverse Weather for Autonomous Driving Through Data Merging and YOLOv8

Debasis Kumar * and Naveed Muhammad

Posted Date: 1 September 2023

doi: 10.20944/preprints202309.0050.v1

Keywords: Autonomous Driving; Harsh Weather; Object Detection; Data Merging; Deep Neural Networks; YOLOv8

*Article*

# Object Detection in Adverse Weather for Autonomous Driving Through Data Merging and YOLOv8

**Debasis Kumar** [1,*] **and Naveed Muhammad** [2]

Current address: Institute of Computer Science, University of Tartu, Narva maantee 18, 51009 Tartu, Estonia; debasis.kumar@ut.ee; naveed.muhammad@ut.ee

* Correspondence: debasis.kumar@ut.ee

**Abstract:** For autonomous driving, perception is a primary and essential element that fundamentally deals with the insight into the ego vehicle's environment through sensors. Perception is a challenging task suffering from dynamic objects and continuous environmental changes. The issue gets worse due to interrupting the quality of perception by adverse weather like snow, rain, fog, night light, sand storm, strong daylight, etc. In this work, we have tried to improve camera-based perception accuracy, such as autonomous driving-related object detection in adverse weather. We proposed the improvement of YOLOv8-based object detection in adverse weather through transfer learning using merged data from various harsh weather datasets. Two prosperous open-source datasets (ACDC and DAWN) and their merged dataset were used to detect primary objects on the road in harsh weather. A set of training weights were collected from training on the individual datasets, their merged version, and several subsets of those datasets according to their characteristics. A comparison between the training weights also occurred by evaluating the detection performance on the above-mentioned datasets and their subsets. The evaluation revealed that using custom datasets for training significantly improves the detection performance compared to the YOLOv8 base weights. And using more images through the feature-related data merging technique steadily increases the object detection performance.

**Keywords:** autonomous driving; harsh weather; object detection; data merging; deep neural networks; YOLOv8

## 1. Introduction

Autonomous driving added an advantage to the autonomous industry by unfolding a new generation of safe and efficient transport systems. The autonomous driving community dramatically improved due to progress in computing technology and cost-efficient sensor production. The level of an autonomous vehicle's autonomy is divided into six levels according to human involvement during its operation [1]. It indispensably depends on the subsystems which satisfy the procedure of autonomy: such as perception, localization, behavior prediction, planning, control, etc. Among these, perception is a vital component of autonomous driving that deals with understanding the ego vehicle's environment using sensors. Then the perception result helps to execute consequent tasks through this information. Due to dynamic objects and ongoing environmental changes, perception is something that can be challenging. Interruptions in perception quality caused by inclement weather such as snow, rain, fog, night light, sand storms, etc., the perception problem gets worse. In this situation, studying weather conditions emanates to fulfilling the weather invariant perception and creates a research community to defeat the shortcomings. Almost every place on the earth experiences different types of seasonal influence on vision and perception, at least alternative day and night conditions in the periodic circle. Due to the sensor's inability in adverse weather, it is ubiquitous to fail to detect lane marks, road marks, landmarks, roadside units, traffic signs, and signals. Environmental perception through sensors is essential for vehicular operation, so it is obvious to study the possible weather variation for the sensors to achieve reliable autonomy.

### 1.1. Objects detection

Among the typical sensors, camera is necessary for perceiving cars, people walking around, and, most importantly, environmental items with colors and signs, such as traffic lights and their colors, traffic signals, road signs, and instructions for driving. However, camera is a device that is least likely to function consistently under varying weather circumstances. Another notable sensor for the contemporary self-driving car is LiDAR (Light Detection and Ranging), which is growing in popularity due to cost reductions brought on by technological advancements. However, it is still expensive and susceptible to weather conditions like snow or smoke. Radar (Radio Detection And Ranging), on the other hand, is more dependable because it is less affected by bad weather [2]. In addition, consistent perception may also benefit from IMU (Inertial Measurement Unit), GNSS (Global Navigation Satellite System), and ultrasonic sensors. Fusion of some of the sensors mentioned above is a popular strategy for enhancing perception. And, sensor fusion is quite helpful in the event of a sensor malfunctioning, particularly in severe weather. Among notable works, [3] proposed an adverse weather dataset (DENSE) using various sensors containing a camera, LiDAR, Radar, gated NIR (near-infrared), and FIR (far-infrared) data in fog, snow, and rain conditions to improve object detection results. Sensor fusion also enhanced detection outcomes in [4,5]. In order to find lanes in bad weather, [4] used GPS (Global Positioning System), LiDAR, and camera data. [5] combined complimentary LiDAR and Radar data using the Multimodal Vehicle Detection Network (MVDNet). The performances and difficulties of various sensors in various weather conditions were covered in more detail through a systematic literature review in [6]. Many studies have combined different sensors for improving perception in good weather without demonstrating how well they perform in bad weather. For instance, [7] used a mix of LiDAR and Radar to detect moving objects reliably, but there was no discussion of performance based on weather. [8] utilized LiDAR and camera data separately and together to compare the performance of pedestrian classification using deep-learning-based early and late multimodal sensor fusion.

Deep learning approaches are replacing traditional perception tasks like object detection, tracking, etc., with newer, more potent ones as a result of the development of machine learning and AI (Artificial Intelligence) technologies. For instance, [9] used probabilistic perception techniques, such as the Rao-Blackwellized particle filter for data association and the Kalman filter for object tracking, among others, for object recognition, tracking, and classification. On the other hand, [10] improved the outcomes in varied weather situations by using a deep learning framework for object detection and tracking, along with a few other techniques. Even though there is no comparison offered between the deep-learning-based technique and other techniques, deep learning techniques are easily expandable by including new ideas (such as, layers modification of the neural networks) for result improvement. There are a few more examples of deep learning-based identification tasks for autonomous driving-related objects like cars, people, traffic lights, drivable pathways or lanes, etc. Deep learning frameworks were utilized by [11] to detect vehicles in foggy conditions. [11] used an attention module to better concentrate on prospective information during feature extraction. [12] offered the ZUT (Zachodniopomorski Uniwersytet Technologiczny) dataset and employed well-known YOLOv3 [13] techniques to identify pedestrians in adverse weather conditions, including rain, fog, frost, etc. There are further examples of the YOLO (You Only Look Once) approach being used to detect pedestrians. For example, [14] used YOLOv3 (and a modified version of it) to detect pedestrians in hazy weather, while [15] used it in regular weather after enhancing the YOLOv2 model to YOLO-R for greater accuracy. In [16–18], more examples of use cases of deep learning networks for object detection in adverse weather conditions were described. [18] used simulated datasets derived from computer simulators to discuss the impact of different weather conditions on sensor data and its impact on obstacle detection. The [16] proposed the dual subnet network (DSNet) trained in an end-to-end manner and jointly learned three tasks; visibility enhancement, object classification, and object localization to execute object detection. [17] proposed an adaptive image enhancement model called the DENet trained using a neural network in an end-to-end manner and added with the YOLOv3 method to obtain DE-YOLO, which improves the detection result compared to the YOLOv3 method. [19] used the DENSE [3] dataset and CNN (Convolutional Neural Network) based sensor fusion for drivable path detection in snowy conditions, i.e., snow-covered roads, and claimed that the drivable path detection might serve as a preparatory step before object detection. Some useful portions of the DENSE dataset were manually processed and labeled before usage because they were not labeled for semantic segmentation.

Both traditional computer vision techniques and deep learning-based frameworks are highly helpful in improving camera-based perception relying on images or videos. Image enhancement, restoration, and dehazing are among the methods that may be used for images or videos to boost vision quality, and they are particularly helpful in enhancing object detection. [20] trained the DriveRetinex in an end-to-end manner containing two subnets, namely Decom-Net and Enhance-Net, for decomposing a color image into a reflectance map and an illumination map, then enhanced the light level in the illumination map. The image-enhancing network improved the object detection results trained on the Low-Light Drive (LOL-Drive) dataset collected by the authors. The Image Adaptive YOLO (IA-YOLO), which combines the YOLOv3 and parameter-predicted convolutional neural network known as CNN-PP, is another image-enhancing method covered in [21]. The IA-YOLO trained in an end-to-end manner and improved the detection performance in the foggy weather and low light scenarios. [22] proposed Gaussian-YOLOv3 by reformating the loss function of the YOLOv3 and additionally predicting the localization uncertainty of the bounding box during object detection. After applying these techniques, the detection results improved by increasing the true positives and reducing the false positives. By dehazing the scene and training a neural network according to their respective settings, [23,24] worked on improving detection performance in hazy weather. [23] trained ReViewNet using a hybrid weighted loss function and looked twice over the hazy images to optimize the dehazing algorithm. The dehazing algorithm was used in [24] by training the BAD-Net in an end-to-end manner that connects the dehazing module and the detection module. The work also discussed the effects of image restoration results and did not use the image restoration result during the training of the networks. According to [25], as demonstrated by [21], the standard image restoration procedure was ineffective in enhancing detection outcomes. However, the authors improved detection performance using image restoration in cloudy and hazy conditions by concentrating on pertinent adversarial attacks [25].

The performance of deep learning frameworks may now be easily improved by increasing the volume of datasets through the arrival of new machines' increased processing power and better storage capacity. The problem of data shortage could be solved by image augmentation, data association, etc., as every neural network method ravens for vast data. These data-growing strategies are becoming more popular in addressing issues like poor detection performance. [26] presents an overall survey on vehicle perception and asserts that the corresponding research community still needs to improve vehicular perception, such as object detection in poor weather, and data fusion could solve this problem. [27] used dataset construction based on the GAN (Generative Adversarial Networks) and cycleGAN architectures that helped to create the seven versions of different weather conditions of a single dataset and created another seven versions of an augmented dataset from that single dataset. These datasets were produced using applicable computer techniques, such as adding fictitious droplets. The approach solved the difficulty of collecting data from the real world. It helped to learn different weather features from versions proposed in datasets, thus improving detection results in various weather conditions. [28] also included artificial droplets to examine performance in low-light weather augmentation for several racing car tracks condition using real-world and simulator data. The detection performance was tested on various weather images containing late afternoon, sunset, dusk, night, and some different size of droplets. However, instead of visual accuracy, the effort mainly concentrated on the real-time performance of the perception subsystem.

Thus far, the discussions above show that several efforts have been made to enhance object detection through various methodologies. Growing data volume by various processes, such as image augmentation, artificial data creation, etc., is one of the more effective methods among them. Recent research on existing autonomous driving datasets and their covered weather aspects for perception was presented in a part of the survey in [2]. According to the table-6 proposed in [2], there are still a few research gaps, such as combining all the different types of harsh weather. As an illustration, consider integrating the three types of weather—snow, nightlight, and strong daylight—into a single dataset. By following the idea from the above mentioned article, our research aims to combine most weather features and analyze how feature combination from different datasets affects object detection. Previously, in addition to the usual weather, several datasets proposed weather circumstances like snow, rain, nightlight, fog, haze, smog, sandstorm, cloud, overcast, etc. The work plan recommends integrating datasets from several sources containing various meteorological variables to detect autonomous vehicle-related objects in a weather-consistent manner. This study will aid in determining the effects of feature accumulation from various datasets, diverse geometrical regions, and meteorological circumstances using multiple data sources.

### *1.2. Relevant datasets*

Deep learning has recently performed outstandingly well in various visual tasks, including scene perception, object identification, object tracking, 3D geometry estimation, image segmentation, and many more tasks pertinent to autonomous vehicles. Pioneering work like ImageNet [29] is one of the finest examples of the deep neural network and dataset proposed on visual recognition. Similarly, many datasets have been proposed to improve visual recognition results in the last decade. More intensely, if we focused on the usefulness of similar type datasets in autonomous driving, then the KITTI [30], Microsoft COCO [31], and Cityscapes [32] datasets contribute largely to adapting the usefulness of visual perception in the autonomous vehicle's field. Other influential datasets which were contributed to the diversity and quantity of resources are Camvid [33], Caltech [34], Daimler-CB [35], CVC [36], NICTA [37], Daimler-DB [38], INRIA [39], ETH [40], TUDBrussels [41], Leuven [42], Daimler Arban Segmentation [43], and many more examples enriched the repository. These datasets were collected from the real world (or synthetic) and used for various purposes, such as pedestrian classification, pedestrian detection, object detection, semantic segmentation, etc.

This study focuses on particular datasets containing various weather characteristics to achieve robust perception in harsh weather. The principal harsh weather components are snow, rain, fog, night, and sand, which are also containing sub-components such as mist, haze, smog, strong daylight, reflective night light, rainy night, rain storm, sand storm, dust tornado, clouds, overcast, sunset, shadow, etc. Some datasets worked to cover these weather characteristics but were limited to a few specific features independently. Therefore, we cannot concede those datasets uniquely for weather invariant perceptions nor ubiquitously considered useful in harsh weather. So, we have chosen some prosperous open datasets according to their feature, usefulness, and weather characteristics and planned to merge them to cover all features and generate a fruitful repository to eliminate incompleteness in environmental characteristics. Moreover, we know that deep learning methods, i.e., neural networks are extremely data-hungry; thus, the fusion of different datasets could be useful for learning useful features globally. There are ample datasets available for harsh weather, such as Radiate [44], EU [45], KAIST multispectral [46], WildDash [47], Raincouver [48], 4Seasons [49], Snowy Driving [50], Waymo Open [51], Argoverse [52], DDD17 [53], D2-City [54], nuScenes [55], CADCD [56], LIBRE [57], Foggy Cityscape [58], etc. These datasets mostly contain camera images (some are with LiDAR, Radar, GPS, and IMU data) taken from the real world, considering various weather characteristics in the real environment. On the other hand, SYNTHIA [59] and ALSD [60] contain synthetic images from a computer-generated virtual world including some adverse weather features. Despite the huge progress in the autonomous driving data field, we have chosen a few particular datasets based on availability, features, geometrical variation, and combinations of more useful weather characteristics. The following datasets were collected online for further progress in this work. The data collection process followed the official data collection rule for the corresponding resources and registered on their websites to take official permission (if required) to use their data in future research.

For this investigation, our attention has been on camera images because the camera is the most crucial sensor for environmental scene perception, especially for traffic sign recognition, object identification, and object localization. On the other hand, LiDAR is also easily influenced by weather. Nevertheless, we can add Radar and IMU as additional sensors, but it is optional to improve those because they are insignificantly influenced by harsh weather. Among the image datasets, the Breakly Deep Drive (BDD) [61] and Eurocity [62] could be useful resources for performing the main contribution to the data merging for this work. Some other datasets may also subsidize with fewer images but very intensive features from the perspective of weather characteristics. The BDD dataset contains 100,000 camera images (collected from driving video) from various cities in the USA, such as New York, Berkeley, and San Francisco. Besides typical weather, it contains images with other weather features like rain, fog, overcast, cloud, snow, and night light. The BDD dataset created a benchmark for performing the particular ten tasks mentioned in their paper, and annotated every image according to their tasks. These are image tagging, lane detection, drivable area segmentation, road object detection, semantic segmentation, instance segmentation, multi-object detection tracking, multi-object segmentation tracking, domain adaptation, and imitation learning. Though the dataset is rich in the perspective of the number of images, it contains fewer images according to harsh weather features. Only 23 fog images, 213 rain images, 765 snow images, and 345 night images are useful to contribute to learning the weather features [63]. So, a manual search was required to elicit those useful images from the huge dataset, which might not be feasible, and it is better to focus on a different useful dataset containing more harsh weather images and feature diversity. Another rich dataset was Eurocity, which contains 47,300 images collected from 31 cities in 12 European countries, characterized by geometrical varieties, covering various weather categories such as rain, fog, snow, and night light, besides normal conditions. But the dataset primarily focused on pedestrian detection in traffic scenes contains 238,200 persons. The Mapillary dataset [64] collected 25,000 street images from the Mapillary street view app. The collection was distributed worldwide based on images taken in the rain, snow, fog, and night, besides natural weather conditions. So, this dataset was the most diverse dataset in geographical extent, which contains various scene perceptions from the world's different geometry related to various traffic rules and road conditions. However, the dataset also has the same problem as the BDD dataset. This work focused on studying the diverse weather conditions, claiming that there should be adequate weather-diverse images compared to typical weather images. Playing for Benchmarks [65], which contained 254,064 high-resolution image frames from the video collection, was the richest dataset regarding the number of images. However, the images were captured from a virtual environment created by computers. The ApolloScape dataset [66] was considered for capturing driving in bright sunlight, as this kind of situation frequently occurs while driving. The dataset contains 143,906 images collected from four regions in China, but a small portion of them are useful for learning adverse situations, as this work targeted. Besides the strong light, driving vehicles also could face sun glare, which can quickly impair vision and result in serious accidents. Until recently, there was a shortage of autonomous driving datasets with images of objects to detect in sun glare, which the autonomous driving research community accidentally neglected. Among the few papers that did so, [67] suggested a glare dataset for detecting traffic signs only. The "Adverse Condition Dataset with Correspondance" dataset [63], also known as the "ACDC dataset", had 4006 camera images from Zurich recorded in four weather conditions: rain, fog, snow, and night. The ACDC has all photos with one of any weather features and 4006 images evenly distributed for each weather characteristic, which was very useful despite having a much smaller number of images than the BDD or Eurocity datasets. Therefore, from the perspective of usefulness, this dataset was more prosperous than the other dataset described previously. The 19 classes provided by Cityscape [32] were annotated on the ACDC dataset using pixel-level semantic segmentation and trustworthy ground truth. The paper tested multiple existing neural networks and compared their performance on the dataset. The "Vehicle Detection in Adverse

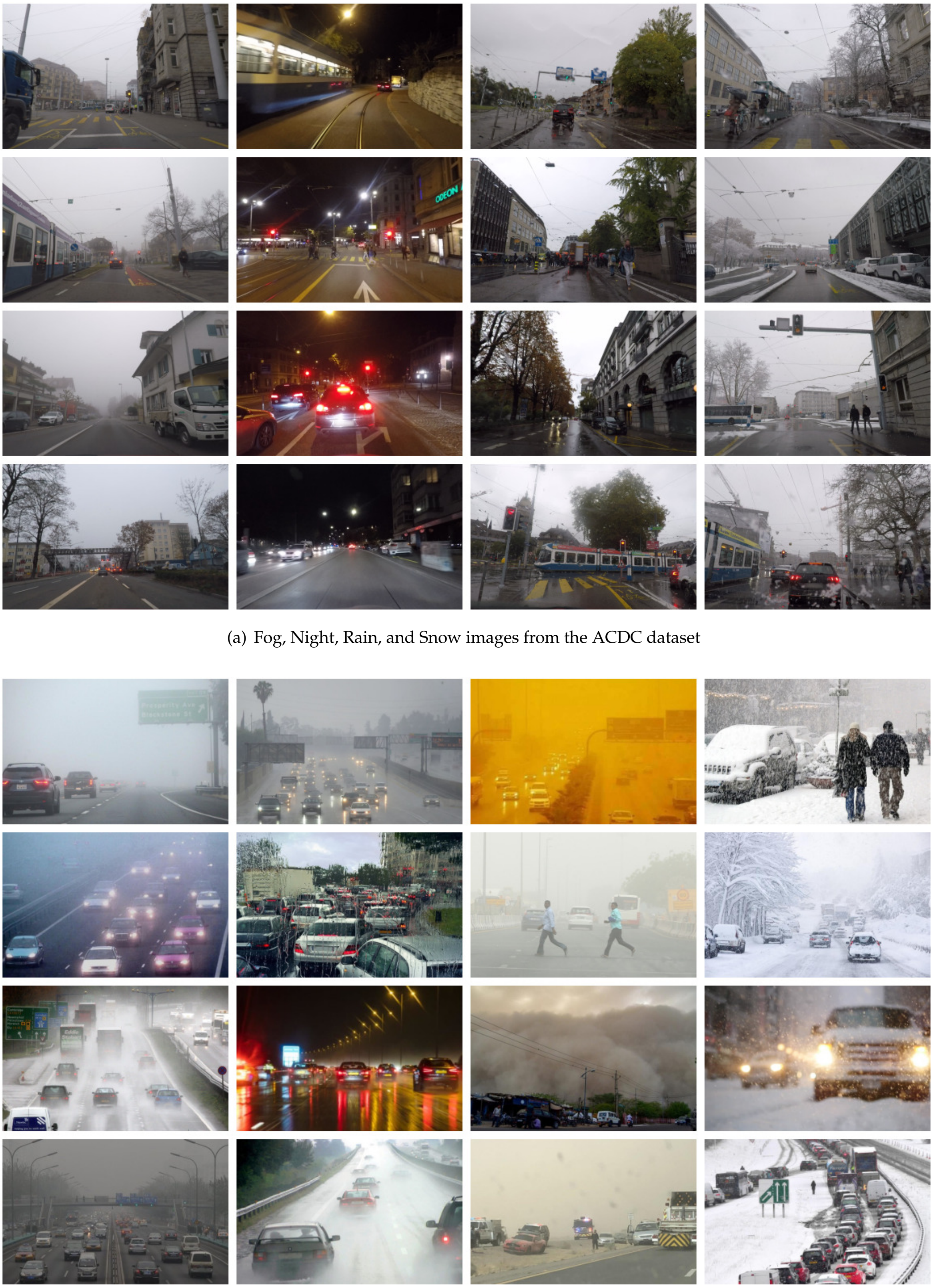
Weather Nature" dataset, also known as the "DAWN dataset" [68], which only contains 1,027 photos gathered from web searches on Google and Bing, was another highly helpful dataset. However, it was selected for its extremely harsh weather qualities, which can serve as a real-world example for training and testing under adverse conditions. It also includes several sandstorm images that offer distinctive aspects compared to other datasets mentioned earlier. Seven thousand eight hundred forty-five bounding boxes for vehicles, buses, trucks, motorcycles, bicycles, pedestrians, and riders were labeled in the DAWN dataset annotated by the LabelMe tool. The ACDC and DAWN datasets' primary distinguishing feature includes every image in adverse weather. So, from the above discussion, the criteria for choosing the datasets are clear now. But, we can collect by manually selecting the relevant images from the abovementioned datasets, which might be time consuming but relevant to extend this work further. However, we discovered that the ACDC and DAWN datasets were the most helpful for our analysis.

### *1.3. Current study*

From the explanation above, we have concluded that the dataset and its components are numerous. Nevertheless, every dataset was annotated according to their requirements and failed to cover all aspects of weather conditions from a unique study. Therefore, choosing a few helpful images and merging them to cover all the weather feature in a single dataset for feeding into a neural network for training was an avaricious aim. And we combined the ACDC and DAWN datasets (some example images are presented in Figure 1), which practically cover all adverse conditions, excluding direct sunshine, i.e., sun glare. Still, the unique and accurate annotation for the combined dataset was the biggest hurdle for this work. It was unable to combine the labeling since the various datasets utilize various methods to annotate their datasets. The goal was to develop a quick and efficient approach for annotating the combined data per the needs of object detection. Before that, knowing each dataset's labeling procedure and strength for network training was useful for subsequent research.

Since this investigation was limited to 2D object detection through image data by camera sensors, the primary proposition of this work was to use the YOLO as an object detection method. As discussed above, a few researchers have used the YOLO method for object detection in autonomous driving. Some tried to modify the method to boost performance (IA-YOLO, YOLO-R, DE-YOLO, Gaussian-YOLOv3, etc.). [69,70] discussed network architectures, challenges, advantages, applications, and many more for different versions of the YOLO method. Recently, [71] published the most updated information about the YOLO algorithm and discussed all the YOLO version releases until the most recent version (YOLOv8, [72]). Our approach is very similar to [73] regarding object detection in bad weather for autonomous driving. [73] suggested training a custom model using the YOLOv5 to detect objects in adverse weather. Just one model was trained in 18 minutes and 12 seconds using 239 images downloaded from the Roboflow website, and it reached an accuracy of about 25%. Comparatively, our work proposed assessing the feature merging between two datasets to open up additional prospects for combining more datasets, and in some cases, achieved more than 90% accuracy utilizing the YOLOv8.

The following list of contributions (state below) can be made by this work in its entirety: a review of the literature and discussion of various object detection techniques related to autonomous vehicles, the selection of specific datasets from a small number of relevant datasets, and the study's overall plan, as stated in Section 1. Data collection, annotation, merging, training, and evaluation approaches are covered in the part of the methodology discussed in Section 2. A comprehensive analysis of the experiments and findings is provided in Section 3. Section 4 discussed the study's limitations and areas for future expansion. Section 5 concludes the work by summarizing the work and providing recommendations for future researchers.



(a) Fog, Night, Rain, and Snow images from the ACDC dataset

(b) Fog, Rain, Sand, and Snow images from the DAWN dataset

**Figure 1.** Example images from the ACDC and DAWN datasets. Column-wise weather features respectively.

## 2. Methodology

The procedures indicated in the next subsection were used to carry out the experiment for this investigation. Following the discussion of the selection process for all relevant datasets, the collection of datasets and the process for data annotation are now briefly covered in the first subsection. Then, a description of data processing and data merging follows. Finally, the training, validating, and testing approaches are discussed with their associated evaluation criteria.

### *2.1. Data collection and annotation*

The two datasets used for this study's primary contribution were open-sourced and widely accessible. The DAWN dataset was primarily used for object detection in harsh weather and was annotated for six classes (car, bus, truck, motorcycle, bicycle, and person). The ACDC dataset was proposed for driving scene understanding in harsh weather through image segmentation. ACDC dataset was widely used for domain adaptation, such as a study of the change of data domain. These datasets were available online, along with their corresponding annotations. But a special labeling procedure was needed to combine them. The labels on the DAWN dataset are incompatible with the most recent versions of YOLO, even though they were created for the YOLOv3 based on Darknet architecture. The DAWN dataset, for instance, used the LabelMe tool to annotate the images, classifying people as label 1, bicycles as label 2, cars as label 3, and so on. Contrarily, the most recent versions of the YOLO, which were trained on the COCO dataset, assigned 0 to people, 1 to bicycles, 2 to cars, etc. Since transfer learning was intended to be used in this study's custom data training, a completely new labeling order assignment is acceptable during the transfer learning. However, we followed a universal labeling process similar to the COCO dataset that is compatible with any version of YOLO trained on the COCO dataset. We planned to detect the first ten objects and keep their corresponding labels as the COCO dataset to make unique annotations for all datasets we intended to merge. Namely the annotation used as 0:person, 1:bicycle, 2:car, 3:motorcycle, 4:airplane, 5:bus, 6:train, 7:truck, 8:boat, and 9:traffic light. Though we do not expect to detect a boat or airplanes while driving on the road, we kept them as rare objects and focused more on detecting objects like vehicles, pedestrians, traffic lights, etc. So, performing a new annotation was required for all images. And after new labeling, the annotation is compatible with the YOLO method to use their weights also. This study contributed to annotating both datasets to detect primary objects (the first ten objects from the COCO dataset) in harsh weather for autonomous driving. Interestingly, before this annotation, according to our knowledge, the ACDC dataset had never been labeled for object detection. The reputable data annotation website makesense.ai [1], assisted with the manual data annotation for this work. For the YOLO method (version 8), the makesense.ai generated labels in text format, and for the PASCAL VOC annotation [2], labels were in HTML (Hypertext Markup Language) format. The annotation method, which took an average of five minutes per image, attempted to incorporate all pertinent objects regardless of size and proximity to the camera. A few photos from the ACDC dataset (comparatively fewer from the DAWN) were deleted from the dataset since they did not contain any targeted objects for detection.

### *2.2. Data processing and merging*

After labeling all the pertinent images from two datasets, data processing was crucial in this work before training. The datasets were prepared in different versions for training, validating, and testing by the YOLOv8 algorithm. The YOLOv8 takes two different versions of image size as input for training, i.e., 640*640 and 1280*1280. Since the YOLO algorithm does not contain any image processing or augmentation process, all the image processing, including data resizing and augmentation, was

[1] https://www.makesense.ai
[2] http://host.robots.ox.ac.uk/pascal/VOC/

performed with the help of the Roboflow website [3], recommended by the YOLOv8 method. The Roboflow website was useful for conducting image resizing and augmentation tasks containing horizontal flip, vertical flip, crop, grayscale, brightness, blur, rotation, shear, hue, saturation, exposure, noise, cutout, mosaic, and many more. The Roboflow website was also helpful in separating all images to train, validate, and test images into 70, 20, and 10 percent ratios. Before merging the datasets, it was crucial to split them into the train, validate, and test sets because failing to do so may result in a new combination of segregated images contaminating the evaluation dataset (test and validation images). The many combinations of relevant data augmentation techniques (crop, blur, etc.) were carried out and handled as multiple enhanced dataset versions. Through the training process, these versions assisted in evaluating the effectiveness of those versions' findings for object detection. The relevant data augmentation mentioned here conveys that, for example, a vertical flip is not useful to this study. Nine versions of resized (640*640) image-augmented datasets were arranged for training. Then the best-augmented version was chosen by the validation result during the training and selected as the final version of the augmented images used here. And the same augmentation settings also followed for the 1280*1280 image-sized version. Different resized versions of the same image were used because they allowed for faster training, and occasionally the training results revealed accuracy that was even better than the original version. So, in this study, training results were presented for different versions of the processed image as follows. Here version one contains the original dataset without augmentation or resized, and version two presents the image-augmented datasets resized by 640*640, version three contains the original datasets (non-augmented) resized by 640*640, version four presents the original datasets resized by 1280*1280, and version five contains augmented datasets resized by 1280*1280. The five versions of the 'MERGED' dataset were created by merging the corresponding training, validation, and testing images from these five versions of the DAWN and ACDC datasets. According to their particular meteorological features, several mergings also occurred between the subgroups of the DAWN and ACDC datasets. The validation and testing images were the same across the same-sized augmented or non-augmented data versions since the data augmentation was only done on the training images. For a more thorough depiction, see Table 1.

**Table 1.** Explanation of train, validation, and test data versions.

| Weights trained on | | | | | Weights evaluated on | |
|---|---|---|---|---|---|---|
| **Weight trained on Version no.** | **Input images** | **Base weights** | **Size of images during training** | **Approximate Training time for the MERGED dataset (32 epochs)** | **Size of validation images** | **Size of test images** |
| V1 | Raw images | yolov8x.pt | 640*640 | 4 hours | Raw images | Raw images |
| V2 | 640*640 (augmented) | yolov8l.pt | 640*640 | 5 hours | 640*640 | 640*640 |
| V3 | 640*640 | yolov8l.pt | 640*640 | 2 hours | 640*640 | 640*640 |
| V4 | 1280*1280 | yolov8x.pt | 640*640 | 3.5 hours | 1280*1280 | 1280*1280 |
| V5 | 1280*1280 (augmented) | yolov8x.pt | 640*640 | 4.5 hours for 16 epochs | 1280*1280 | 1280*1280 |

[3] https://roboflow.com

### 2.3. Training and evaluation

We employed Python programming based on the Google Colab service for training and result evaluation through validation and testing. Until this study was done, the latest version of the YOLO was the YOLOv8. This deep-learning-based neural network model is faster and gains better accuracy compared to previous versions in object detection. This study used the YOLOv8 algorithm and its pre-trained weights as a backbone for training on custom data through transfer learning. Like other YOLO versions, six key attributes are responsible for object detection with a bounding box. The key attributes are the x and y coordinates of the top left corner of the bounding box, the width and height of the bounding box, a confidence score with a probability between zero to one, and class ID. From the performance perspective, the YOLOv8 was already a perfect algorithm for object detection, and this work helped improve detection accuracy in harsh weather. The YOLOv8 GitHub repository [72] assisted in setting up training on custom data and other associated works (saving the training model, accuracy checking, etc.). Six versions of pre-trained weights could be used as a base for transfer learning during training for custom data. The base weights are 'yolov8n.pt' for nano-objects detection, 'yolov8s.pt' for short objects, and so on for medium and large, and 'yolov8x.pt' for extra large objects. Accuracy gained by training on these various base weights was also considered to choose the best weight for further training.

Since all the images did not contain objects, they were removed from the respective folder of both datasets. The ACDC dataset was divided into 2715 training images, 770 validation images, and 383 test images. So, only 3868 images were used among the 4006 images proposed by the original datasets. The DAWN dataset was divided into 700 training images, 203 validation, and 100 test images. During the accuracy evaluation, it is important to remember that the same number of images and the same images inside the validation and test set should be present to evaluate and compare the results. Table 2 presents more details about the image distribution.

**Table 2.** Image distribution.

| | DAWN | | | | ACDC | | | | MERGED | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Weather** | **Train** | **Valid** | **Test** | **Total** | **Train** | **Valid** | **Test** | **Total** | **Train** | **Valid** | **Test** | **Total** |
| **Sand** | 223 | 63 | 33 | 319 | * | * | * | * | 223 | 63 | 33 | 319 |
| **Fog** | 193 | 59 | 27 | 279 | 638 | 179 | 88 | 905 | 831 | 238 | 115 | 1184 |
| **Rain** | 142 | 40 | 19 | 201 | 698 | 198 | 98 | 994 | 840 | 238 | 117 | 1195 |
| **Snow** | 142 | 41 | 21 | 204 | 700 | 200 | 100 | 1000 | 842 | 241 | 121 | 1204 |
| **Night** | * | * | * | * | 679 | 193 | 97 | 969 | 679 | 193 | 97 | 969 |
| **Total** | 700 | 203 | 100 | **1003** | 2715 | 770 | 383 | **3868** | 3415 | 973 | 483 | **4871** |

Every different version of the dataset was trained for 32 epochs and resized to 640*640 during training without depending on their input data dimension. And all other hyperparameters are kept at their default values for the YOLO method. The training performance was evaluated on corresponding validation data after finishing the training process (result displayed after every training). These results helped to understand the performance of the merged and associated data versions. Then the training weights were saved and evaluated the performance on test sets. So, the weights were first performed on the corresponding validation data during training and then tested on the corresponding test set. Then the saved weights were also performed on the test and validation set of other data versions. The detailed workflow is shown in Figure 2.

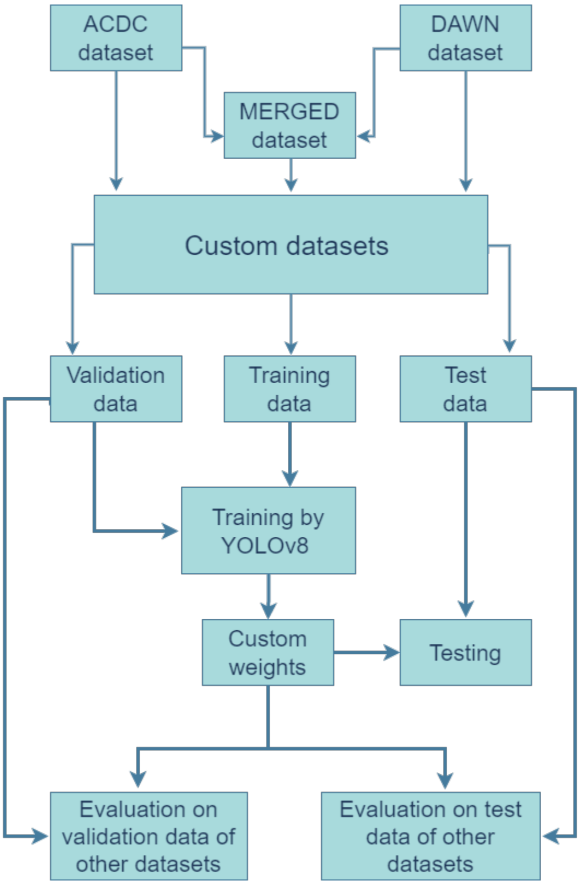


**Figure 2.** Object detection model using the YOLOv8 algorithm.

## 3. Experiments and Results Discussion

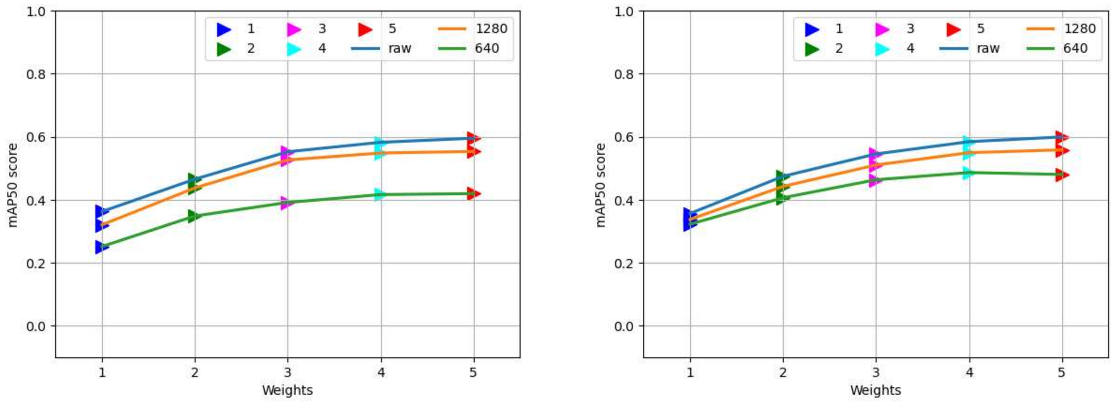
All the experiments were conducted in this work using Python under the scope of the Google Colab. The YOLOv8 algorithm helped to perform training using transfer learning on top of its base weights and custom dataset. The ACDC and DAWN datasets and their merged dataset as MERGED were used as the custom datasets. After training, the saved weights of the different versions of training datasets were used to evaluate the performance using the various versions of the validation and test set. The object detection results were evaluated using the mAP (mean average precision) in two different outcomes (mAP50 and mAP50-95) predefined by the YOLOv8 algorithm. The IoU (Intersection Over Union) was measured as the bounding box overlap between the ground truth and predicted bounding box, and mAP50 considered the corresponding detection as true positive where the IoU is greater than 0.5. Similarly, mAP50-95 used all different thresholds between 0.5 and 0.95 using step 0.05. This work usually used mPA50 for presenting results, and rarely mAP50-95, only where it was mentioned. The sequence of experiments was arranged first to choose the best base weight for training. The YOLO algorithm does not contain any image augmentation technique and assigns the augmentation part to a third party like Roboflow. We used Roboflow to generate different augmented versions of the datasets, then chose the best image augmentation version based on their validation result during training. In the next step, we evaluated training performance on the ACDC and DAWN datasets separately. We then used their merged dataset to train and evaluate the performance of merged data and two base datasets.

The training process of the YOLOv8 can take any version of images independent of size but converts the training images into 640*640 or 1280*1280 before feeding them into the training network. But, resizing them earlier before feeding them into the network is useful for saving training time. As mentioned before, this work studied five versions: raw images (without resizing), two resized versions

(1280*1280 and 640*640), and their augmented versions named raw, 1280, 640, augmented 1280, and augmented 640, respectively (see Table 1). All five versions were used as input but were resized to 640*640 during training. We intended to train on the 1280*1280 version besides the smaller version, but the Google Colaboratory failed to manage the bigger version of the images during training due to GPU power limitation. So, there is a research gap in using a powerful computer to train the dataset as image size 1280 during training beside input as the same size. Nevertheless, we used the five versions as input despite resizing to 640 by the algorithm during training to compare performances and their training time.

### *3.1. Choosing the best weights and augmentation*

Figure 3 shows the object detection (mAP score) performance of YOLOv8's existing weights on validation and test set of the MERGED images, respectively. The weight number one to five (along the x-axis) refers to the base weights for detecting nano (yolov8n.pt), small (yolov8s.pt), medium (yolov8m.pt), large (yolov8l.pt), and extra-large (yolov8x.pt) objects correspondingly. The three versions of MERGED datasets (raw and two resized versions, since the test and validation images were the same for augmented and non-augmented data) are presented to evaluate the detection performance of based weights on the MERGED dataset. From the figure, it is clear that the extra-large weight performed the best object detection result on any version of images without depending on image sizes. The large and medium weights also performed well compared to the nano and small versions of the YOLOv8's weight. It is worth mentioning that all the weights perform below 0.6 at mAP50. Figure 3 helped choose the best base weight to train the custom datasets further via transfer learning. According to the result, we chose the extra-large weight (yolov8x.pt) as the base for the raw images, 1280*1280-sized images and their augmented version. Contrarily the large weight (yolov8l.pt) was used for the 640*640-sized images and their augmented version as input images.



(a) Performance on the validation images

(b) Performance on the test images

**Figure 3.** Performance of YOLOv8's default weights on valid and test images of the 'MERGED' dataset.

Figure 4 presents the performance of object detection on the validation set of the ACDC dataset by different versions of image augmentation of the ACDC dataset. First, nine different sets of augmented images were arranged from the ACDC train images and trained to evaluate the performances of the corresponding augmented version on the ACDC validation images. Since the results are almost similar for all the versions, it is not worth mentioning the augmentation technique used by each augmented version for choosing the best augmentation result. Version zero, denoted in A0, is defined as no augmentation at all. On the other hand, version A7 contains all types of augmentation together, using horizontal flip, crop (0 to 20% zoom), rotation (-15 to 15 degrees), shear (up to 15 degrees in both horizontal and vertical), and all the following up to 25% (grayscale, hue, saturation, exposure, blur, and brightness (both the darken and brighten)). So, all the augmentation together was not useful for better performance; even version A0 (without augmentation) performed better than any augmentation for the ACDC dataset. After examining the performance of various image augmentation, we decided to use version A8 as the augmented version for other datasets, where horizontal flip and cropping were used as augmentation tools.

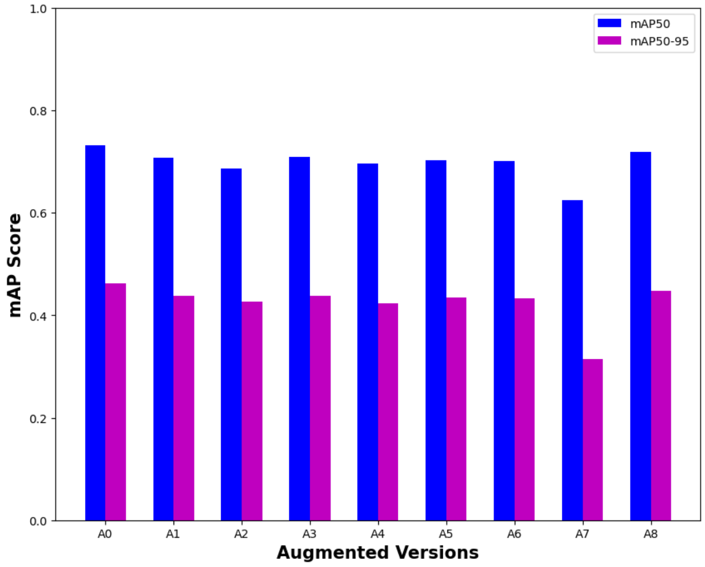


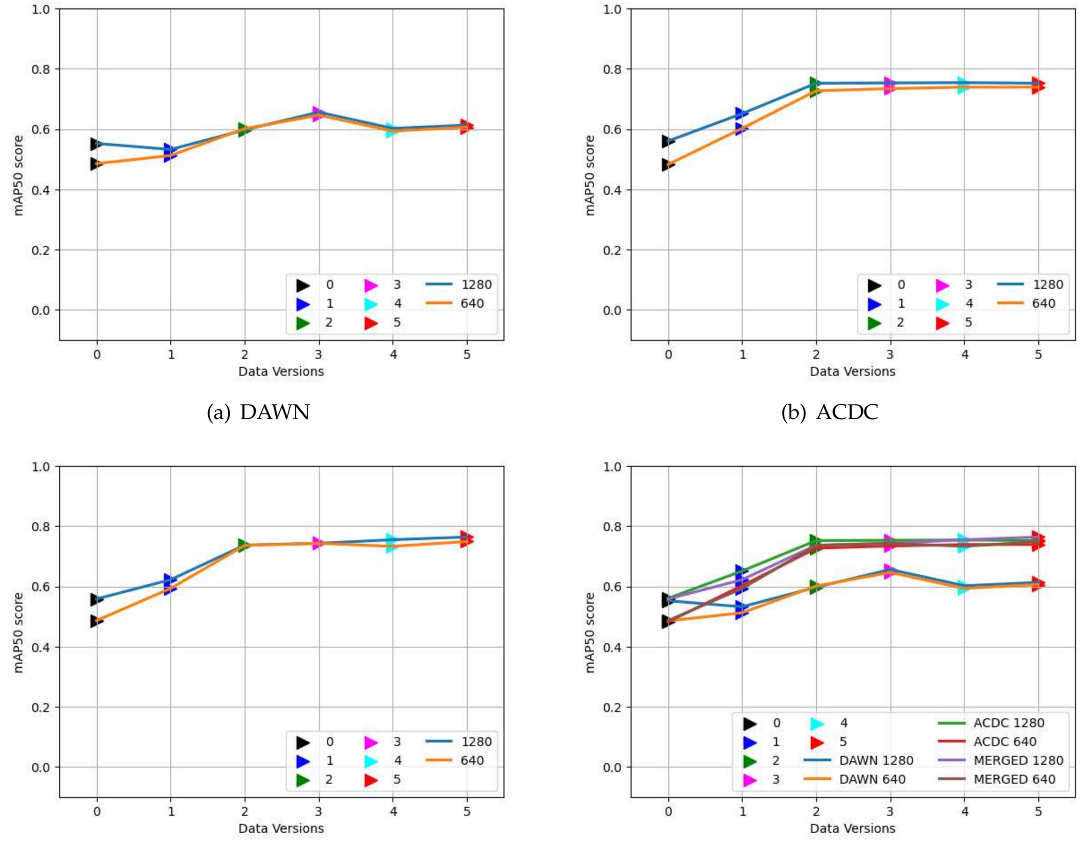
**Figure 4.** Detection performance of different versions of image augmentation.

### *3.2. Evaluation of weights training on the DAWN, ACDC, and MERGED dataset*

Figure 5 displays the detection performance of training weights of five different versions of the 5(a) DAWN, 5(b) ACDC, and 5(c) MERGED datasets on two different sizes of test images of the corresponding dataset, Figure 5(d) presents all three results together. Version zero (black) mentioned here displays the performance of the YOLO algorithm's base weight (yolov8x.pt). The other nodes presented trained on raw images (the original dataset as version one) in blue, augmented 640 (version two) in green, without augmented 640 (version three) in magenta, without augmented 1280 (version four) in cyan, and augmented 1280 (version five) in red.

From all the first three figures (5(a), 5(b), 5(c)), it is clear that training on the corresponding dataset improves the result of detection compared to the base weight of the YOLOv8 method. Even training on the resized images perform well compared to the raw images. Since the algorithm resized the training images into 640*640 during training independent of input sizes, the effects of different sizes of input images are almost the same for both small and large versions, even for augmented and non-augmented versions, but exceptionally better than training on raw images. The bigger images as a test set (1280, presented in navy) perform slightly better than smaller images (640, presented in orange) during testing. Therefore, we chose the best size (1280*1280) for evaluating valid and test images from this result. Since weights were produced using resized images during training, it is preferable to resize images before utilizing them for evaluation and subsequent use because different datasets may contain images of different sizes.

Finally, Figure 5d shows all three results together for comparing the training performance of mentioned weights on the corresponding dataset. Training on the ACDC dataset was better at detecting objects in its test images than the DAWN dataset. This figure also helped to identify the accuracy elevation by training on custom data uplifted to near 0.8 from below 0.6. The MERGED dataset contains more images than the ACDC dataset, and the ACDC contains more images than the DAWN. We can conclude that the number of feature related unique images in training data was the reason for better performance if the merged data performed better on every subset, which we found later by the following results.



(a) DAWN

(b) ACDC

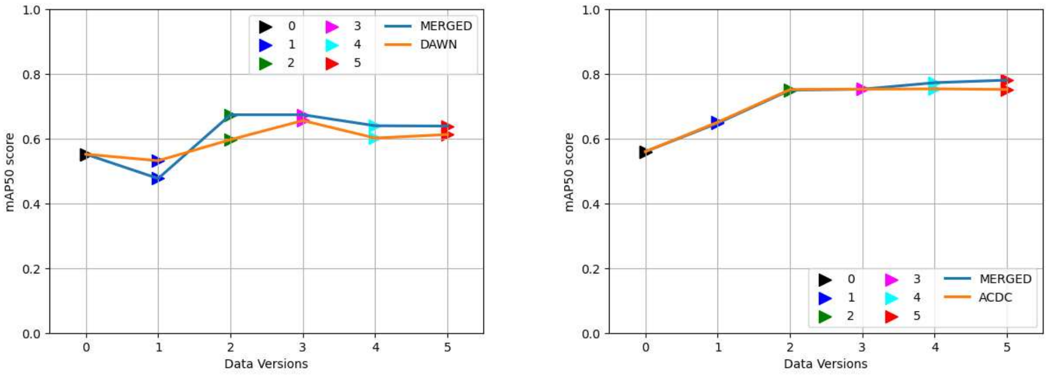
(c) MERGED

(d) All together

**Figure 5.** Performance of training weights using the DAWN, ACDC, and MERGED dataset on their corresponding test images.

### *3.3. Effects of the MERGED dataset on the DAWN and ACDC dataset*

Figure 6 shows the elevation in detection performance after merging the two datasets compared to the corresponding single dataset on the test set of the single dataset. The result is the mAP50 score of the different versions of training weights collected from the different versions of the MERGED dataset and compared to the training weights of the individual dataset. Both weights were performed on the test images of the individual datasets. First, 6(a) compares results trained on the DAWN and the merged dataset (MERGED) over the DAWN test images. Similarly, 6(b) presents the comparison between the training weights of ADCD dataset and the MERGED dataset over the ACDC test images. Since the ACDC dataset contains approximately four times more images, the result of the DAWN dataset benefitted more from the merged dataset in their detection results. In Figure 6b, the performances of the weights are almost similar except for version five input, so adding the DAWN dataset affects less on improving the detection of the ACDC images. However, 6(a) shows a significant elevation in performance after adding the ACDC dataset compared to only the DAWN dataset. So, adding more number of images or merging more datasets could improve detection results further. Additionally, in Figure 6a, training on the raw dataset (version one) shows an impressive result: merging different datasets of various sizes could harm training results. And pre-resized input improves results for both the datasets and their merged version. However, resizing them before training also saves training time compared to the non-resized version (Table 1).

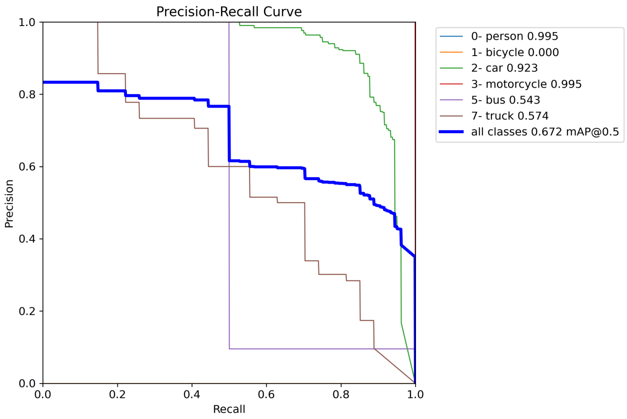


(a) DAWN

(b) ACDC

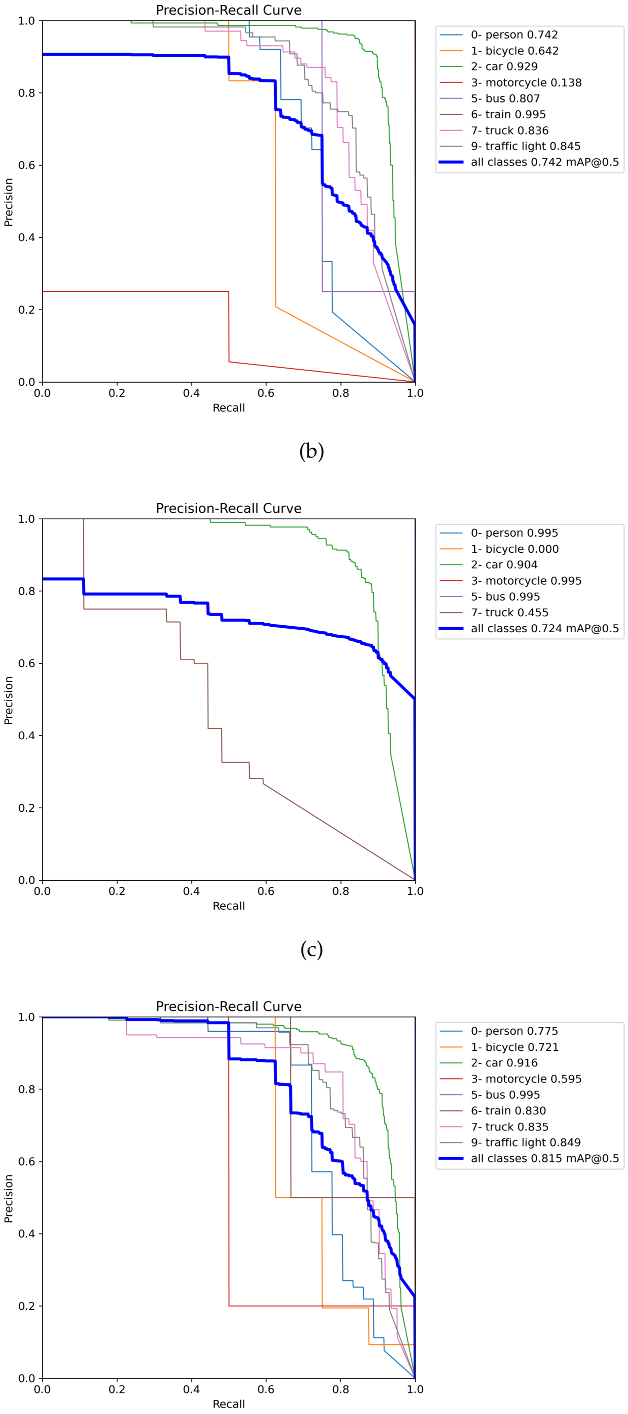
**Figure 6.** Performance of training on the MERGED dataset, and (a) the DAWN dataset on DAWN test images, (b) the ACDC dataset on ACDC test images.

Figure 7 presents the precision-recall (PR) curves using training weights of all the merged images (MERGED) (in 8(g) and 8(h)), trained on the individual DAWN and ACDC (respectively in 8(e) and 8(f)), only merged between fog images of the ACDC and DAWN (in 8(c) and 8(d)), and corresponding fog images of the DAWN in 7(a) and ACDC in 8(b). The PR curves were evaluated only on the fog test images, where the left column presents the PR curves of the DAWN dataset, and the right column presents the ACDC dataset. From the left column, 7(a) presents the mAP50 score of 0.672 evaluated on the DAWN fog test data using weight training on the DAWN fog data. In 8(c), the detection result improved for the weight training on the merged fog (merging between both the ACDC fog and DAWN fog) up to 0.724, and finally, the detection results improved to 0.75 for the weight training on the MERGED training images (in 8(g)). However, the detection result performed by weight trained on the DAWN dataset scored 0.704 (in 8(e)), which is comparatively less, but this case is rare and is still congruent with the relative results. Nevertheless, in the right column, detection performed by training on the ACDC fog images gained the mAP50 of 0.742 on the ACDC fog test images (in 8(b)), which was improved to 0.815 for the weight training on the merged fog images (in 8(d)). The results further improved to 0.884 for the weight trained on the ACDC dataset (in 8(f)) and boosted to mAP50 of 0.91 for the weight training on all the merged (MERGED) train images (in 8(h)). So, object detection results in fog significantly improve by adding more images (feature-related) from different datasets and weather conditions.



(a)

**Figure 7.** *Cont.*

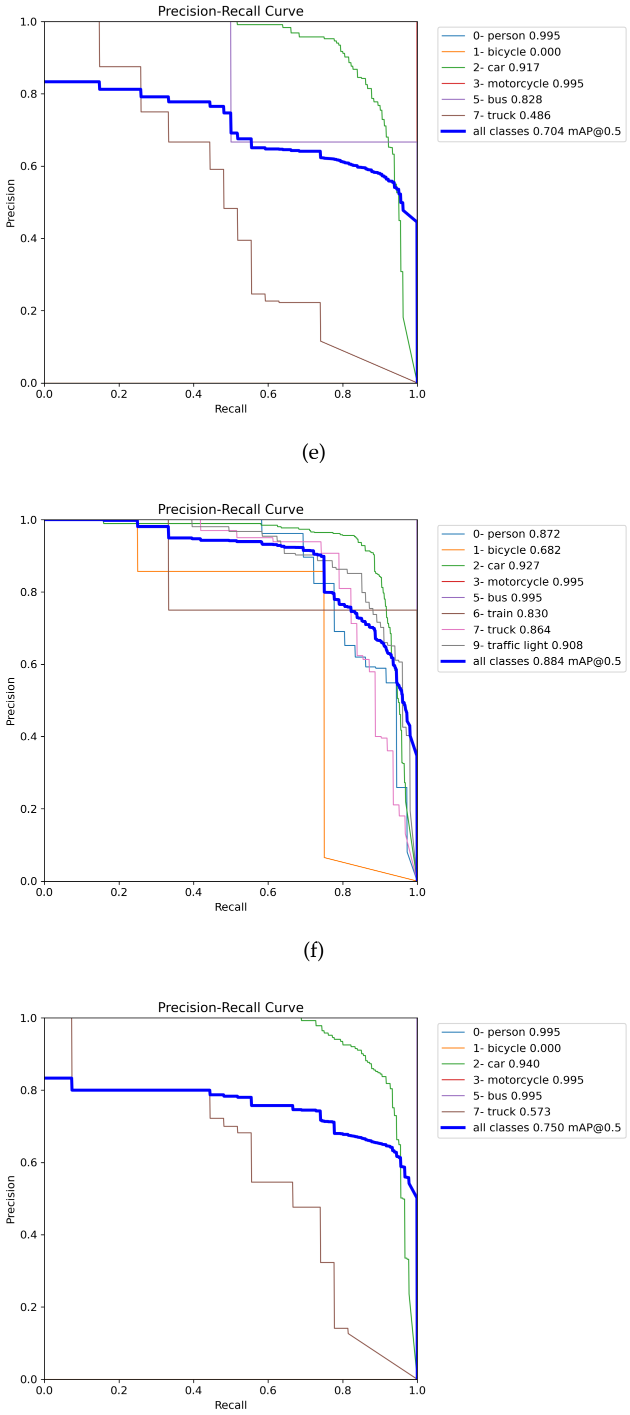


(b)

(c)

(d)

**Figure 7.** *Cont.*

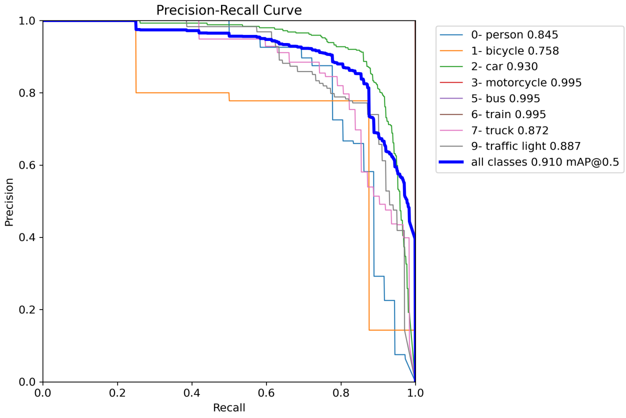


(e)

(f)

(g)

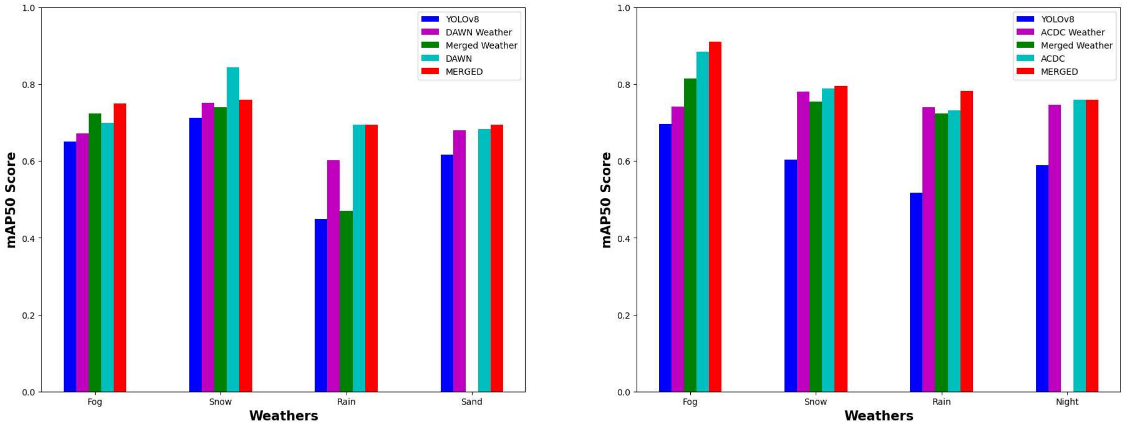
**Figure 7.** *Cont.*



(h)

**Figure 7.** Precision-Recall (PR) curves for the Fog data.

Similarly, we have two more common weather features between the ACDC and DAWN datasets, such as rain and snow. And two individual weather features like sand images for the DAWN and night images for the ACDC dataset only. We are skipping the discussion part of PR curves for those weathers to avoid redundancy. Instead, we have used Figure 8 to compare the mAP50 scores of different weathers and data versions. The result shows a comparison between training weights' detection scores on test images of individual weather characteristics separated into two parts for the DAWN and ACDC datasets, respectively. Each weather characteristic of the ACDC graph shows a comparison between the detection performance of corresponding base weights and trained on the "ACDC weather", "merged weather", "ACDC", and "MERGED", respectively (in Figure 8b). And in a similar way for the DAWN dataset also (in Figure 8a). Weight trained on MERGED mentioned here implicates trained on all the merged training images. In contrast, "merged weather" means merging only the corresponding weather between the two datasets. Similarly, the "DAWN weather" or "ACDC weather" mean "DAWN fog", "ACDC rain", etc., depending on the corresponding weather labels presented through x-axis.



(a) DAWN

(b) ACDC

**Figure 8.** Gradual improvement of the object detection result by incorporating more images through the feature-related data merging technique.

The bar graphs (Figure 8) depict a common pattern of improving detection scores by adding more feature-related images. A steady increase in the detection score was clearly revealed for every trained weight compared to the base result (shown in blue). The results shown in purple are used for weights training on particular weather from the corresponding dataset, which improves performance compared to the base weights, and in some cases, performs better than merged weather (shown in green). The weights trained on the merged weather are missing for both the sand and night images due to the fact that they are not present in both datasets together. Though they (green bar) improved results for the fog images compared to the individual weathers but did not perform well for the rain and snow images. These are examples of a few exceptions in this study, which will be investigated later through the limitation part (section 4). The weights trained on the ACDC and DAWN datasets are used to depicted by cyan bars to show the improvement of detection compared to weights trained on data subsets (merged or individual weathers) discussed previously. Finally, the results using weight trained on the MERGED dataset portrayed in red outperform every result trained on previously discussed data subsets.

However, there are a few exceptions, such as for the DAWN rain images, the detection result drop unexpectedly for the weight trained on the merged rain dataset. This can be explained through feature redundancy, i.e., sometimes adding features could harm some particularly learned features of a small subset due to feature redundancy. But the cases are rare, or Fig. (9) of the limitation part of this study [4] could explain more. Still, despite a few sudden rises or falls appeared in the results, Figure 8 is evident enough to show gradual improvement of the object detection result by incorporating more images through the feature-related data merging technique.

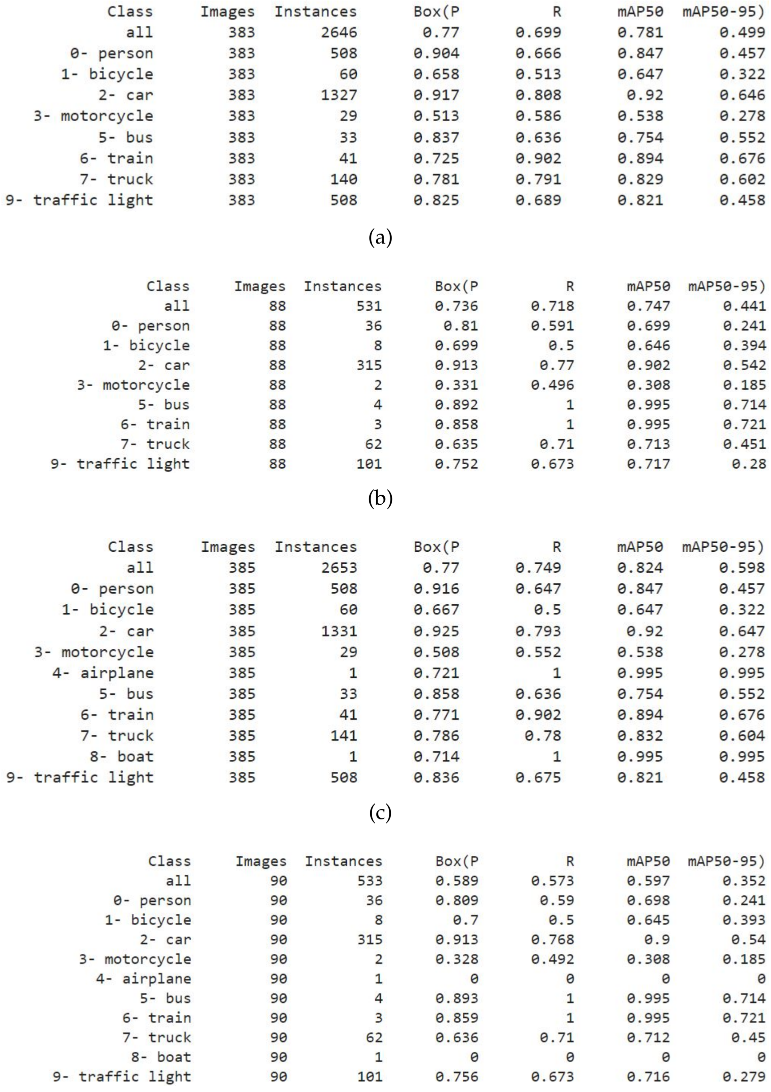
## 4. Limitations and future works

This study was conducted under the limitation of the free use of Google Colaboratory. Due to the time limitation of the GPU uses of the Google Colab, every training performed here used the same number of epochs to compare their accuracy. The regular epoch size of every training was 32, but the augmented 1280 (version five) used 16 epochs for training. This study also intended to use the bigger version (image size 1280*1280) as input and then use the same size during training also. But the bigger images were resized to 640*640 by the YOLO algorithm during training. Due to GPU power limitation, it could not execute as 1280*1280 image size during training. So, there is a research gap in improving accuracy by training on the bigger version with powerful computational opportunities.

Another limitation of this study comes from datasets and their annotation format. Both datasets are annotated from scratch to make them compatible together and with the YOLO method. The YOLO algorithm (version 8) was trained on the COCO dataset as the backbone, and the annotation of this study used the first ten objects in the same order. So, although the DAWN images were annotated in the YOLO format for fewer objects with different annotation orders, this study required manual annotations to correct the order and add a few more objects to detect. So, this study is limited to comparing with previous results studied by other researchers. But now, an extension of this study can use the current annotation [4] for the DAWN and ACDC datasets to improve the accuracy, which is also compatible with the YOLO weights, but adding more useful features and datasets requires compatible annotation. The annotation of this study tried to include all the objects, even very small objects present in the far distance, to make detection earlier for warning about the presence of objects in harsh weather. So, accuracy might be less compared to a different study that uses a new annotation. Another thing to mention is that the ACDC dataset is collected from Zurich, where the Bus, Train, and Tram look similar, so it is confusing to learn a perfect feature for differentiating them. However, during annotating the ACDC images, the 'Tram' is treated as a 'Train' here. Now, by considering these limitations of the annotation process, an extension of the current study is possible using the annotation files shared in the GitHub repository (footnote 4) as a base and adding more relevant datasets to improve detection results further.

[4] https://github.com/DebasisKumar21/Labels.git

Figure 9 shows a weakness in the result computation metric of the YOLO algorithm. Figure 9a,c show an accuracy difference by contaminating the datasets with just two images. This study used only eight objects to detect among the first ten objects of the COCO dataset, i.e., not trained to detect a boat or an airplane. Still, their place in the object list was assigned even though no instance was present in the dataset. The contamination of the dataset by two images that contain at least a boat and an airplane in the training and testing sets (same images so can detect them) improved the result unexpectedly by detecting those objects. Only seven instances are added from the two additional images from 9(a) to 9(c). The detection results for all the other objects are the same, but the results improved from 0.781 to 0.824 by detecting only one boat and airplane. Contrarily, from 9(b) to 9(d), two instances were added by two additional images (improving 531 objects to 533). The weight failed to detect two instances (a boat and an airplane) since no boat or airplane was present in the studied datasets. The result fell from mAP50 of 0.747 to 0.597 for missing to detect those particular two objects. For this study, there are approximately 6996 objects present in the validation images and 3492 in the test images. So, manipulating the accuracy table by removing or contaminating images with rare objects is easy. However, this study used a fixed number of images and the same images for every test and validation set to perform the comparison fairly.



```
            Class     Images  Instances      Box(P          R      mAP50  mAP50-95)
              all        383       2646       0.77      0.699      0.781      0.499
        0- person        383        508      0.904      0.666      0.847      0.457
       1- bicycle        383         60      0.658      0.513      0.647      0.322
           2- car        383       1327      0.917      0.808       0.92      0.646
    3- motorcycle        383         29      0.513      0.586      0.538      0.278
           5- bus        383         33      0.837      0.636      0.754      0.552
         6- train        383         41      0.725      0.902      0.894      0.676
         7- truck        383        140      0.781      0.791      0.829      0.602
 9- traffic light        383        508      0.825      0.689      0.821      0.458
```

(a)

```
            Class     Images  Instances      Box(P          R      mAP50  mAP50-95)
              all         88        531      0.736      0.718      0.747      0.441
        0- person         88         36       0.81      0.591      0.699      0.241
       1- bicycle         88          8      0.699        0.5      0.646      0.394
           2- car         88        315      0.913       0.77      0.902      0.542
    3- motorcycle         88          2      0.331      0.496      0.308      0.185
           5- bus         88          4      0.892          1      0.995      0.714
         6- train         88          3      0.858          1      0.995      0.721
         7- truck         88         62      0.635       0.71      0.713      0.451
 9- traffic light         88        101      0.752      0.673      0.717       0.28
```

(b)

```
            Class     Images  Instances      Box(P          R      mAP50  mAP50-95)
              all        385       2653       0.77      0.749      0.824      0.598
        0- person        385        508      0.916      0.647      0.847      0.457
       1- bicycle        385         60      0.667        0.5      0.647      0.322
           2- car        385       1331      0.925      0.793       0.92      0.647
    3- motorcycle        385         29      0.508      0.552      0.538      0.278
      4- airplane        385          1      0.721          1      0.995      0.995
           5- bus        385         33      0.858      0.636      0.754      0.552
         6- train        385         41      0.771      0.902      0.894      0.676
         7- truck        385        141      0.786       0.78      0.832      0.604
          8- boat        385          1      0.714          1      0.995      0.995
 9- traffic light        385        508      0.836      0.675      0.821      0.458
```

(c)

```
            Class     Images  Instances      Box(P          R      mAP50  mAP50-95)
              all         90        533      0.589      0.573      0.597      0.352
        0- person         90         36      0.809       0.59      0.698      0.241
       1- bicycle         90          8        0.7        0.5      0.645      0.393
           2- car         90        315      0.913      0.768        0.9       0.54
    3- motorcycle         90          2      0.328      0.492      0.308      0.185
      4- airplane         90          1          0          0          0          0
           5- bus         90          4      0.893          1      0.995      0.714
         6- train         90          3      0.859          1      0.995      0.721
         7- truck         90         62      0.636       0.71      0.712       0.45
          8- boat         90          1          0          0          0          0
 9- traffic light         90        101      0.756      0.673      0.716      0.279
```

(d)

**Figure 9.** Limitation of accuracy.

## 5. Conclusion

In this study, we proposed the use of combined data from several severe weather datasets for training through transfer learning to enhance YOLOv8-based object detection in bad weather. We used two effective open-source datasets (DAWN and ACDC) to identify important roadside objects in severe weather. First, the datasets were collected from the corresponding websites and annotated in the YOLOv8 format for the first ten objects of the COCO datasets. The datasets contain weather features of fog, rain, snow, night, and sand, and the individual weather images were divided into 70% train images, 20% validation images, and 10% test images. Then, they were merged to separately combine train, valid, and test images to create a MERGED data version. Various data augmentations were also used to choose the best-augmented version according to their detection performance. Images were resized to various versions to check their performance and training time. These data versions were used to train custom weights and test their object detection performance on test images. The performances on validation images were also achieved after finishing the training process through the accuracy table produced by the YOLO algorithm.

The proposed data merging technique improved the object detection accuracy significantly compared to the performance of the base weights of the YOLOv8 algorithm. The results compared the performance of weights training on the individual DAWN and ACDC datasets, their merged dataset (MERGED), and their distinct weather subsets. The results presented via graphs (Figure 6) show that the MERGED dataset performed better than weights training on the individual datasets. The accuracy improvement presented in the bar graphs (Figure 8) shows that training on a custom dataset improves the object detection results further, and the accuracy was elevated with the addition of more images (with relevant data features). Noticeably, the training weight collected from training on the MERGED datasets performed best on every subset of the relevant dataset, gradually becoming better than training on a particular subset after merging those subsets. So, this study concludes that merging more diverse images of feature-relevant datasets could do better for object detection.

These findings provide the following research-related insights. The detection outcomes could be enhanced even more by starting with the base datasets and labels presented here and adding more datasets to them. A more powerful computer may further enhance the outcomes by using larger image sizes during training or training for more epochs. The meteorological features utilized here cover nearly every adverse weather situation, but some environmental factors, such as sun glare, are still absent. In addition to severe weather, more images from datasets with regular weather could lead to more accurate object detection also.

**Acknowledgments:** This research has been financed by the European Social Fund via "ICT programme" measure.

## References

1. Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles (J3016B), 2018, [online] Available: https://www.sae.org/standards/content/j3016_201806/.
2. Y. Zhang, A. Carballo, H. Yang, and K. Takeda, "Autonomous driving in adverse weather conditions: A survey," arXiv [cs.RO], 2021.
3. M. Bijelic et al., "Seeing through fog without seeing fog: Deep multimodal sensor fusion in unseen adverse weather," in 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.
4. U. Lee et al., "EureCar turbo: A self-driving car that can handle adverse weather conditions," in 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2016.
5. K. Qian, S. Zhu, X. Zhang, and L. E. Li, "Robust multimodal vehicle detection in foggy weather using complementary lidar and radar signals," in 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2021.
6. A. S. Mohammed, A. Amamou, F. K. Ayevide, S. Kelouwani, K. Agbossou, and N. Zioui, "The perception system of intelligent ground vehicles in all weather conditions: A systematic literature review," Sensors (Basel), vol. 20, no. 22, p. 6532, 2020.

7. B. Yang, R. Guo, M. Liang, S. Casas, and R. Urtasun, “RadarNet: Exploiting radar for robust perception of dynamic objects,” in Computer Vision - ECCV 2020, Cham: Springer International Publishing, 2020, pp. 496-512.
8. G. Melotti, C. Premebida, N. M. M. da S. Goncalves, U. J. C. Nunes, and D. R. Faria, “Multimodal CNN pedestrian classification: A study on combining LIDAR and camera data,” in 2018 21st International Conference on Intelligent Transportation Systems (ITSC), 2018.
9. P. Radecki, M. Campbell, and K. Matzen, “All weather perception: Joint data association, tracking, and classification for autonomous ground vehicles,” arXiv [cs.SY], 2016.
10. M. Hassaballah, M. A. Kenk, K. Muhammad, and S. Minaee, “Vehicle detection and tracking in adverse weather using a deep learning framework,” IEEE Trans. Intell. Transp. Syst., vol. 22, no. 7, pp. 4230-4242, 2021.
11. N. Tao, J. Xiangkun, D. Xiaodong, S. Jinmiao, and L. Ranran, “Vehicle detection method with low-carbon technology in haze weather based on deep neural network,” Int. J. Low-Carbon Technol., vol. 17, pp. 1151-1157, 2022.
12. P. Tumas, A. Nowosielski, and A. Serackis, “Pedestrian detection in severe weather conditions,” IEEE Access, vol. 8, pp. 62775-62784, 2020.
13. J. Redmon and A. Farhadi, “YOLOv3:An incremental improvement,” arXiv: 1804.02767, 2018.
14. G. Li, Y. Yang, and X. Qu, “Deep learning approaches on pedestrian detection in hazy weather,” IEEE Trans. Ind. Electron., vol. 67, no. 10, pp. 8889-8899, 2020.
15. W. Lan, J. Dang, Y. Wang, and S. Wang, “Pedestrian detection based on YOLO network model,” in 2018 IEEE International Conference on Mechatronics and Automation (ICMA), 2018.
16. S. C. Huang, T. H. Le, and D. W. Jaw, “DSNet: Joint semantic learning for object detection in inclement weather conditions,” IEEE Trans. Pattern Anal. Mach. Intell., pp. 1-1, 2020.
17. Qin, Qingpao, Kan Chang, Mengyuan Huang, and Guiqing Li. “DENet: Detection-driven Enhancement Network for Object Detection under Adverse Weather Conditions.” In Proceedings of the Asian Conference on Computer Vision, pp. 2813-2829. 2022.
18. R. Song, J. Wetherall, S. Maskell, and J. Ralph, “Weather effects on obstacle detection for autonomous car,” in Proceedings of the 6th International Conference on Vehicle Technology and Intelligent Transport Systems, 2020.
19. N. A. Rawashdeh, J. P. Bos, and N. J. Abu-Alrub, “Drivable path detection using CNN sensor fusion for autonomous driving in the snow,” in Autonomous Systems: Sensors, Processing, and Security for Vehicles and Infrastructure 2021, 2021.
20. L. H. Pham, D. N. N. Tran, and J. W. Jeon, “Low-light image enhancement for autonomous driving systems using DriveRetinex-net,” in 2020 IEEE International Conference on Consumer Electronics - Asia (ICCE-Asia), 2020.
21. W. Liu, G. Ren, R. Yu, S. Guo, J. Zhu, and L. Zhang, “Image-Adaptive YOLO for object detection in adverse weather conditions,” arXiv [cs.CV], 2021.
22. J. Choi, D. Chun, H. Kim, and H. J. Lee, “Gaussian YOLOv3: An accurate and fast object detector using localization uncertainty for autonomous driving,” in 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 2019.
23. A. Mehra, M. Mandal, P. Narang, and V. Chamola, “ReViewNet: A fast and resource optimized network for enabling safe autonomous driving in hazy weather conditions,” IEEE Trans. Intell. Transp. Syst., vol. 22, no. 7, pp. 4256-4266, 2021.
24. C. Li et al., “Detection-friendly dehazing: Object detection in real-world hazy scenes,” IEEE Trans. Pattern Anal. Mach. Intell., vol. PP, 2023.
25. Sun, S., Ren, W., Wang, T., and Cao, X. (2022). Rethinking Image Restoration for Object Detection. Advances in Neural Information Processing Systems, 35, 4461-4474.
26. J. Van Brummelen, M. O'Brien, D. Gruyer, and H. Najjaran, “Autonomous vehicle perception: The technology of today and tomorrow,” Transp. Res. Part C Emerg. Technol., vol. 89, pp. 384-406, 2018.
27. V. Musat, I. Fursa, P. Newman, F. Cuzzolin, and A. Bradley, “Multi-weather city: Adverse weather stacking for autonomous driving,” in 2021 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW), 2021.

28. I. Fursa et al., "Worsening perception: Real-time degradation of autonomous vehicle perception performance for simulation of adverse weather conditions," SAE Int. J. Connect. Autom. Veh., vol. 5, no. 1, pp. 87-100, 2022.
29. J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2009, pp. 248-255.
30. A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in 2012 IEEE Conference on Computer Vision and Pattern Recognition, 2012.
31. T. Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollar, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in Proc. Eur. Conf. Comput. Vis., 2014, pp. 740-755.
32. M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2016, pp. 3213-3223.
33. G. J. Brostow, J. Shotton, J. Fauqueur, and R. Cipolla, "Segmentation and recognition using structure from motion point clouds," in Lecture Notes in Computer Science, Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 44-57.
34. P. Dollar, C.Wojek, B. Schiele, and P. Perona, "Pedestrian detection: An evaluation of the state of the art," IEEE Trans. Pattern Anal. Mach. Intell., vol. 34, no. 4, pp. 743-761, Apr. 2012.
35. S.Munder and D.M. Gavrila, "An experimental study on pedestrian classification," IEEE Trans. Pattern Anal. Mach. Intell., vol. 28, no. 11, pp. 1863-1868,Nov. 2006.
36. D. Geronimo, A. Sappa, A. Lopez, and D. Ponsa, "Adaptive image sampling and windows classification for on-board pedestrian detection," in Proc. 5th Int. Conf. Comput. Vis. Syst., 2007, vol. 39.
37. G. Overett, L. Petersson, N. Brewer, L. Andersson, and N. Pettersson, "A new pedestrian dataset for supervised learning," in Proc. IEEE Intell. Veh. Symp., 2008, pp. 373-378.
38. M. Enzweiler and D. M. Gavrila, "Monocular pedestrian detection: Survey and experiments," IEEE Trans. Pattern Anal. Mach. Intell., vol. 31, no. 12, pp. 2179-2195, Dec. 2009.
39. N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2005, vol. 1, pp. 886-893.
40. A. Ess, B. Leibe, and L. Van Gool, "Depth and appearance for mobile scene analysis," in Proc. IEEE Int. Conf. Comput. Vis., 2007, pp. 1-8.
41. C. Wojek, S. Walk, and B. Schiele, "Multi-cue onboard pedestrian detection," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2009, pp. 794-801.
42. B. Leibe, N. Cornelis, K. Cornelis, and L. Van Gool. Dynamic 3D scene anlysis from a moving vehicle. In (CVPR), 2007.
43. T. Scharwächter, M. Enzweiler, U. Franke, and S. Roth. Efficient multi-cue scene segmentation. In (GCPR), 2013.
44. M. Sheeny, E. De Pellegrin, S. Mukherjee, A. Ahrabian, S. Wang, and A. Wallace, "RADIATE: A radar dataset for automotive perception in bad weather," in 2021 IEEE International Conference on Robotics and Automation (ICRA), 2021.
45. Z. Yan, L. Sun, T. Krajnik, and Y. Ruichek, "EU long-term dataset with multiple sensors for autonomous driving," in 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2020.
46. Y. Choi et al., "KAIST multi-spectral day/night data set for autonomous and assisted driving," IEEE Trans. Intell. Transp. Syst., vol. 19, no. 3, pp. 934-948, 2018.
47. O. Zendel, K. Honauer, M. Murschitz, D. Steininger, and G. F. Domínguez, "WildDash - creating hazard-aware benchmarks," in Computer Vision - ECCV 2018, Cham: Springer International Publishing, 2018, pp. 407-421.
48. F. Tung, J. Chen, L. Meng, and J. J. Little, "The raincouver scene parsing benchmark for self-driving in adverse weather and at night," IEEE Robot. Autom. Lett., vol. 2, no. 4, pp. 2188-2193, 2017.
49. P. Wenzel et al., "4Seasons: A cross-season dataset for multi-weather SLAM in autonomous driving," in Lecture Notes in Computer Science, Cham: Springer International Publishing, 2021, pp. 404-417.
50. Y. Lei, T. Emaru, A. A. Ravankar, Y. Kobayashi, and S. Wang, "Semantic image segmentation on snow driving scenarios," in 2020 IEEE International Conference on Mechatronics and Automation (ICMA), 2020.
51. P. Sun et al., "Scalability in Perception for Autonomous Driving: Waymo Open Dataset," in 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.

52. M. F. Chang et al., "Argoverse: 3D tracking and forecasting with rich maps," in 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019.
53. J. Binas, D. Neil, S. C. Liu, and T. Delbruck, "DDD17: End-to-end DAVIS driving dataset," arXiv [cs.CV], 2017.
54. Z. Che et al., "D2-City: A large-scale dashcam video dataset of diverse traffic scenarios," arXiv [cs.LG], 2019.
55. H. Caesar et al., "nuScenes: A Multimodal Dataset for Autonomous Driving," in 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.
56. M. Pitropov et al., "Canadian Adverse Driving Conditions dataset," Int. J. Rob. Res., vol. 40, no. 4-5, pp. 681-690, 2021.
57. A. Carballo et al., "LIBRE: The Multiple 3D LiDAR Dataset," arXiv [cs.RO], 2020.
58. C. Sakaridis, D. Dai, and L. Van Gool, "Semantic foggy scene understanding with synthetic data," Int. J. Comput. Vis., vol. 126, no. 9, pp. 973-992, 2018.
59. G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez, "The SYNTHIA dataset: A large collection of synthetic images for semantic segmentation of urban scenes," in 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.
60. D. Liu, Y. Cui, Z. Cao, and Y. Chen, "A large-scale simulation dataset: Boost the detection accuracy for special weather conditions," in 2020 International Joint Conference on Neural Networks (IJCNN), 2020.
61. F. Yu et al., "BDD100K: A diverse driving dataset for heterogeneous multitask learning," arXiv [cs.CV], 2018.
62. M. Braun, S. Krebs, F. Flohr, and D. Gavrila, "EuroCity Persons: A novel benchmark for person detection in traffic scenes," IEEE Trans. Pattern Anal. Mach. Intell., vol. 41, no. 8, pp. 1844-1861, 2019.
63. C. Sakaridis, D. Dai, and L. Van Gool, "ACDC: The Adverse Conditions Dataset with Correspondences for semantic driving scene understanding," arXiv [cs.CV], 2021.
64. G. Neuhold, T. Ollmann, S. R. Bulo, and P. Kontschieder, "The mapillary vistas dataset for semantic understanding of street scenes," in 2017 IEEE International Conference on Computer Vision (ICCV), 2017.
65. S. R. Richter, Z. Hayder, and V. Koltun, "Playing for Benchmarks," arXiv [cs.CV], 2017.
66. X. Huang, P. Wang, X. Cheng, D. Zhou, Q. Geng, and R. Yang, "The ApolloScape open dataset for autonomous driving and its application," arXiv [cs.CV], 2018.
67. N. Gray et al., "GLARE: A dataset for Traffic Sign detection in sun glare," arXiv [cs.CV], 2022.
68. M. A. Kenk and M. Hassaballah, "DAWN: Vehicle detection in adverse weather nature dataset," arXiv [cs.CV], 2020.
69. P. Jiang, D. Ergu, F. Liu, Y. Cai, and B. Ma, "A review of Yolo algorithm developments," Procedia Comput. Sci., vol. 199, pp. 1066-1073, 2022.
70. T. Diwan, G. Anirudh, and J. V. Tembhurne, "Object detection using YOLO: challenges, architectural successors, datasets and applications," Multimed. Tools Appl., vol. 82, no. 6, pp. 9243-9275, 2023.
71. J. Terven and D. Cordova-Esparza, "A comprehensive review of YOLO: From YOLOv1 to YOLOv8 and beyond," arXiv [cs.CV], 2023.
72. G. Jocher, A. Chaurasia, and J. Qiu, "YOLO by Ultralytics." https://github.com/ultralytics/ultralytics, 2023.
73. T. Sharma, B. Debaque, N. Duclos, A. Chehri, B. Kinder, and P. Fortier, "Deep learning-based object detection and scene perception under bad weather conditions," Electronics (Basel), vol. 11, no. 4, p. 563, 2022.

**Disclaimer/Publisher's Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.