Impact of Phase Data Augmentation on Performance and Robustness in Object Detection

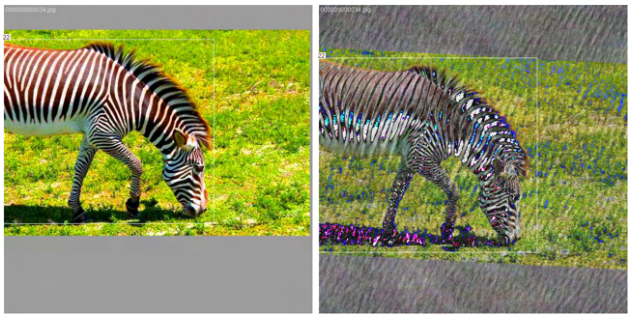
Sytze de Witte
University of Twente, The Netherlands

Abstract

When an image is converted to the Fourier domain, it can be represented by two components: amplitude and phase. Inherently, phase data captures object shapes, which are crucial for object detection. Therefore, applying data augmentation in the phase domain has the potential to enhance object detection models in ways unmatched by traditional augmentation methods. However, despite significant successes in image classification, the application of phase data augmentation in object detection is sparse. This research investigates the effectiveness of phase data augmentation methods in enhancing object detection performance and robustness. Additionally, a novel method applying phase data augmentation selectively within bounding boxes (BBoxes) is introduced, which reduces computational overhead by only applying the costly FFT operation on an area within bounding boxes. Results show that integrating phase data augmentation with default pipelines achieves slightly higher mean average precision (mAP) in general and improves robustness against noise corruptions, though limitations were observed under blur and other miscellaneous corruptions. It was also observed that phase data augmentation had the greatest effect on performance when applied to simpler models on smaller datasets. Future research directions include optimizing hyperparameters, exploring complementary augmentation strategies, and further refining BBox-specific augmentations for enhanced performance and robustness.

1. Introduction

In recent years, convolutional neural networks (CNNs) have become significantly more robust, in part through the application of frequency analysis [3, 19, 20]. More specifically, the importance of the phase spectrum of an image for a robust vision system has become clear. Phase information inherently has the ability to encapsulate shapes of objects, as opposed to amplitude information, which primarily conveys textural details [3, 15]. The approach of applying data augmentation using phase information has yielded notable successes in improving classification accuracy across both standard and corrupted datasets [3, 11, 20].



(a) Regular training image. (b) With phase data augmentation.

Figure 1. Illustrative depiction of the effects of phase data augmentation (VIPAug-G) on object detection. The gray bands are caused by the *LetterBox* transformation, which adds padding to the training images to adjust them to the correct resolution without losing any information.

Despite these advances in the use case of image classification, the exploration of frequency analysis within the context of object detection remains sparse. Object detection, which is a critical component in applications such as autonomous driving, necessitates high precision due to its direct impact on safety and also operational efficiency. The ability of phase data to capture shapes is particularly relevant for object detection, where having more diverse shape representations through phase data augmentation could enhance the model's ability to detect and localize objects accurately, and to do so in adverse conditions. Therefore, the potential of phase data augmentation in this area is intriguing, given its proven benefits in classification tasks.

This project aims to bridge this research gap by investigating the applicability and effectiveness of phase data augmentation in the realm of object detection. The research assesses whether proven phase data augmentation methods, originally used in image classification, can improve object detection performance, especially in challenging conditions typical of real-world applications, such as adverse weather conditions, with examples provided in Appendix A.

1.1. Research question

Following the identified research gap, this study aims to research the performance of object detection models after applying phase data augmentation. To achieve this, the following overarching research question is posed:

- How can the performance of varying object detection models be enhanced by using phase data augmentation methods?

In order to fully explore this research question, it is divided into the following three sub-questions:

1. How do current phase data augmentation methods, as used in image classification, affect the performance of object detection models compared to visual data augmentation methods?
2. Among existing phase and visual data augmentation methods, which one, or what combination thereof, maximizes the performance of object detection models?
3. Are there potential refinements (such as selective augmentation) to current phase data augmentation methods that would improve the performance of object detection models?

2. Related work

2.1. Object detection

Over the last decade, deep learning has greatly improved object detection by learning directly from data, rather than relying on preprogrammed rules or manually crafted features [17]. Object detection architectures can be broadly categorized into two main groups: CNN-based and transformer-based approaches. CNN-based methods can be further divided into one-stage and two-stage architectures. One-stage (mostly anchor-free) models are known for their simplicity and speed, while two-stage (mostly anchor-based models) achieve higher accuracy by employing a region proposal mechanism. In contrast, transformer-based models leverage attention mechanisms to enhance detection performance, representing a more recent innovation in object detection.

One-stage detectors are generally faster than two-stage detectors, as they perform object classification and bounding-box prediction in a single pass through the network. This makes them more suitable for real-time applications, and also the architecture is simpler. When these detectors operate without predefined anchor boxes, they are referred to as anchor-free, which simplifies the model and reduces the reliance on hyperparameters. For this category, **YOLOv8** [7] has been selected as a representative model, as it was the most recent stable version of the most popular one-stage anchor-free detector at the time of setting up this research [18].

In contrast, two-stage detectors focus on accuracy and precision. These models first generate region proposals (potential object locations) and then classify those regions into object categories, typically using anchor boxes to guide the proposal stage. Faster R-CNN [16] is a prominent example in this category, known for its anchor-based approach that effectively balances detection quality with computational efficiency. Although Faster R-CNN was initially considered for this research, it was ultimately not used. Instead, the focus shifted to models optimized for real-time performance, namely YOLOv8 and RT-DETR, where gains in efficiency and robustness can make a more significant impact in practical applications that demand speed, than in models that already prioritize accuracy.

Lastly, there are the transformer-based object detectors, which are currently state-of-the-art and achieve the highest average accuracy on benchmark datasets, and most of the new object detection models released are transformer-based [1]. Originally used for NLP tasks, transformer-based object detectors process an entire image as a sequence of patches, using self-attention mechanisms to weigh the importance of different parts of the image for identifying and localizing objects with high precision. A disadvantage is that currently most transformer-based object detectors cannot meet real-time detection requirements [1]. Nevertheless, this research will include **RT-DETR** (Real Time DEtection TRansformer) [23], since it is designed specifically to address the limitations of real-time performance in transformer-based models. It maintains the high accuracy characteristic of transformer architectures while optimizing for faster inference, making it suitable for real-time applications. Furthermore, the model is included in the Ultralytics *pip* package, enabling the reuse of the code implementing the phase data augmentation methods in YOLOv8, thus simplifying integration and comparisons between models.

2.2. Data augmentation

2.2.1 Traditional data augmentation

Data augmentation has been an essential and widely used technique to improve the performance of all sorts of models by artificially increasing the diversity of training data, to prevent overfitting, thus improving generalization performance and robustness. Early approaches to data augmentation were relatively simple, including geometric transformations like rotation, scaling, translation, and flipping of images, i.e. visual augmentations [8].

As object detection tasks became more complex, with the need to identify smaller, occluded, or highly varied objects within images, more sophisticated data augmentation techniques were developed. Recent advancements have introduced more complex augmentation methods, such as ran-

dom erasing [24], mixup [22], which involve altering image regions to create new training samples. These methods not only increase the variability of the training data, but also encourage models to learn more robust feature representations [6, 21].

2.2.2 Phase data augmentation

In this report, phase data augmentation is defined as any data augmentation method designed to encourage the model to rely more on the phase component of the input data, including approaches that explicitly alter the phase information to achieve this goal. Recent advancements in using phase information to improve the robustness and generalization of CNNs are highlighted in the following studies. Amplitude-Phase Recombination (**APR**) fixes the phases and replaces the amplitudes with those of other images [3]. The method consists of two main parts: APR-P and APR-S. APR-P involves recombining the phase spectrum of one image with the amplitude spectrum of another, thus creating new training samples while maintaining the original phase information to promote robustness. The amplitude from a different image is used as a distracter. APR-S, on the other hand, applies the recombination within a single image using various transformations, ensuring that only self-contained phase and amplitude information are used. Similarly, **FACT** also fixes the phases but mixes the amplitudes with those of other images to create new samples that aid in learning more robust feature representations [19]. **VIPAug** focuses on vital phases that contain domain-invariant features, by selectively manipulating these phases to enhance model robustness across different domains [11]. The method consists of VIPAUG-G, which adds finite variations sampled from a zero-mean Gaussian distribution to vital phases, and VIPAUG-F, which vital phases with those from fractal images, enhancing robustness to amplitude and phase fluctuations.

2.3. Common corruptions

Object detection systems often operate in unpredictable and challenging environments. These conditions introduce various common corruptions that can significantly degrade the performance of these systems. Common corruptions include, but are not limited to, weather conditions such as fog, rain, and snow, as well as other environmental factors like lighting changes, blurring due to motion or out-of-focus cameras, and occlusions.

Studies have shown that these corruptions can adversely affect the accuracy of object detection models. [10] investigated the impact of adverse weather conditions on object detection related to autonomous driving, using YOLOv8, highlighting the need for robust detection methods that can handle such variability. Furthermore, [12] demonstrated the importance of domain adaptation in object detection in foggy weather conditions, further underscoring the susceptibility of detection models to environmental corruptions.

(a) No data augmentation. (b) Default data augmentation.

Figure 2. Representative example of a training image with the default data augmentation method applied.

Lastly, [14] provides an essential reference point for evaluating object detection models under various image corruptions through its easy-to-use *Robust Detection Benchmark*. This benchmark offers tools to generate corrupted datasets, such as COCO-C, by applying distortions like noise, blur, and weather effects to datasets such as PASCAL VOC and COCO. The study reveals that conventional models experience significant performance drops, maintaining only 30% to 60% of their original performance depending on the severity of the corruptions.

3. Methodology

3.1. Data augmentation methods

This subsection outlines the motivations and proposed implementations for the data augmentation methods that are selected to potentially enhance object detection performance, grouping them into default (Figure 2) and phase data (Figure 3) augmentation methods. As can be seen in (Figure 2a), only the *LetterBox* transformation was applied in the experiments with no data augmentation, to ensure consistent image sizes across all experiments.

3.1.1 Default data augmentation methods

The default data augmentation pipelines of YOLOv8 and RT-DETR were used as default augmention baselines for comparison with the state of the art. These pre-configured pipelines use a variety of traditional transformations to introduce variability in object positioning, scaling, and lighting:

- *Mosaic* combines four images into a single composite, providing varied contexts and backgrounds for training ($p = 1.0$).

- *RandomPerspective* applies random scaling (between 50% and 150%) and subtle translations (maximally 10% of image width) to create perspective variations.
- *RandomHSV* modifies hue (±0.015), saturation (±0.7), and brightness (±0.4) to add lighting variability.
- *RandomFlip* flips images horizontally to introduce symmetry in the dataset ($p = 0.5$).
- *Albumentations* adds diverse transformations to the augmentation pipeline ($p = 0.01$).
- *LetterBox* resizes images while maintaining their aspect ratio by adding padding ($p = 1.0$).

These augmentations are standard in YOLOv8 and RT-DETR, already offering robust data diversification and improving model generalization. They serve as a more comprehensive baseline against which the impact of phase data augmentation methods can be assessed.

3.1.2 Phase data augmentation methods

Amplitude-Phase Recombination (APR-S). APR-S was the original phase data augmentation method, motivated by "the powerful generalizability of the human", stating that reducing the dependence on the amplitude spectrum and enhancing the ability to capture the phase spectrum can improve the robustness of a model [3]. The generated samples force the object detection model to pay more attention to the structured information from phase components and keep robust to the variation of the amplitude. For simplicity, APR for the Single Sample (APR-S) has been chosen, since it is less memory complex and easier to implement. Furthermore, should this approach yield promising results, it would be likely that APR-P and APR-SP would also be effective.

For each image in the training set, there is a $p = 0.5$ probability that a Fourier transform is performed on the image (I) and a visually augmented version of that image ($\widetilde{I}$), Then the amplitude component of I is be recombined with the phase component of $\widetilde{I}$ and vice versa, and then returning one of the two recombinations with a 50% probability.

Vital Phase Augmentation (VIPAug-G). VIPAug was the most recently created augmentation method at the time of setting up this research, introducing vital phases. VIPAug encourages the model to depend on the phases over the amplitudes, specifically on the vital phases [11]. It identifies vital phase components that are crucial for maintaining the semantic integrity of images, by performing a 3D discrete Fourier transform (DFT) on the input images to extract amplitude and phase spectrums. Vital phases are determined by examining the amplitudes, with larger amplitudes indicating a higher presence of domain-invariant features. Using a filter, these vital phases are identified across the spatial and channel domains, creating a set of coordinates for each vital phase. The parameters σ_{vital} and σ_{nonvital} denote the standard deviation for the Gaussian distributions for vital and non-vital phases, respectively. For this research, only VIPAug-G was chosen for its flexibility and independence from specific external images, allowing a straightforward integration and effective phase perturbation through Gaussian distribution instead of random fractal images. Similar to APR-S, each training image has a $p = 0.5$ probability of being augmented with VIPAug-G.

(a) APR-S (b) VIPAug-G

Figure 3. Representative examples of a training image with a phase data augmentation method applied.

Impact on models. Phase data augmentation primarily influences the backbone of object detection models, as it alters the structural and shape-related features of the input images in the frequency domain. This leads to changes in the feature maps extracted by the backbone, which are then used by the head for bounding box prediction and classification. While the backbone directly processes the augmented data, the head benefits indirectly through improved feature representations. Although YOLOv8 and RT-DETR might share similarities in their backbones (C2f and HGBlock respectively), their distinct head architectures (grid-based in YOLOv8 and attention-based in RT-DETR) could result in differing sensitivities to phase data augmentation.

Bounding box correction. Recombining the amplitude and phase domains introduced the challenge of determining how and when to apply bounding box correction. This issue arose because the phase data augmentations operate by combining two separate domains, where different augmentations that can either be spatial (e.g., translation or rotation) or non-spatial (e.g., solarize or equalize), may be independently applied to each domain. This can result in inconsistencies between the transformed image and the corresponding bounding box annotations. Bounding box correction ensures that detections remain accurate when spatial augmentations shift or distort object positions. However, in non-spatial transformations, which modify image appearance without altering object locations, correction is unnecessary. The application of different augmentations to am-

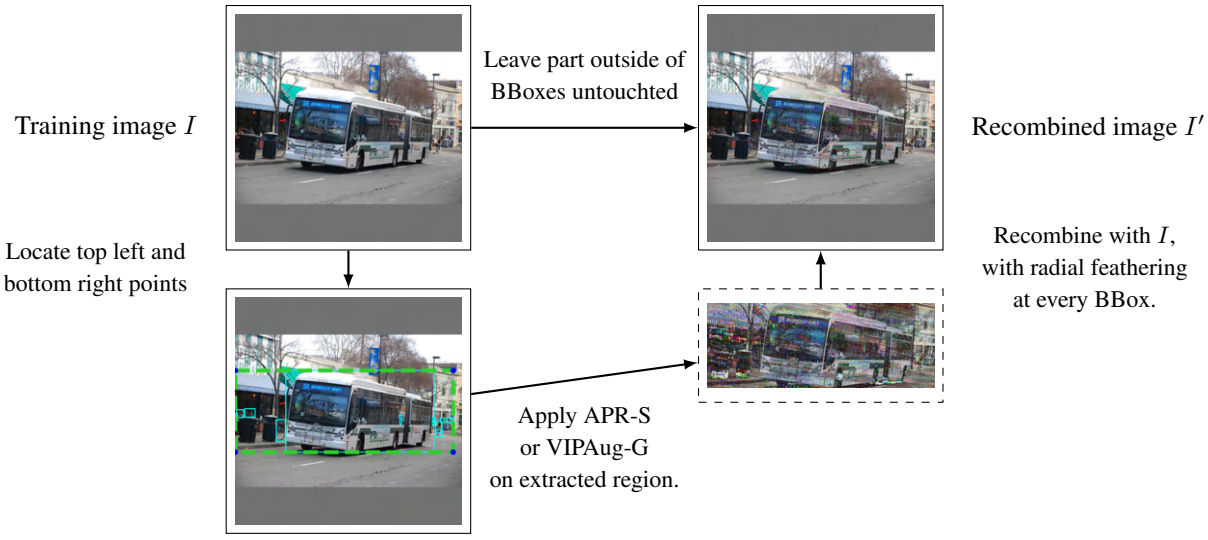


Figure 4. A visualization of regional Bbox augmentation on a training image I. A mask is formed from the top left to the bottom right most bounding box area. Therefore the "costly" FFT operation will be performed on a smaller area, and will include fewer background information. Afterwards, the augmented part is recombined with I to form I', using radial feathering in order to blend the borders.

plitude and phase domains can lead to scenarios where one domain's augmentation affects object placement while the other does not, necessitating selective correction. Based on empirical observations, the following guideline can be established for when to apply bounding box correction in Appendix B. During the implementation process, an improvement of 1 to 2 percentage points in mAP was observed on COCO using a YOLOv8 model. As a result, bounding box correction was used in all subsequent experiments.

3.2. BBox phase data augmentation

The current phase data augmentation methods are made for image classification, and therefore are applied on images as a whole. However, in object detection, the training data contains predefined bounding boxes. Therefore, it is possible to apply different intensities of augmentation to the phase information per label, for example outside of the boxes much more than inside, or vice versa. The strategy chosen for implementing this region-specific augmentation is as follows. The training image is segmented into two masks, inside a bounding box and outside a bounding box. Then phase data augmentation is applied from the top left most bounding box to the bottom right most bounding box in the inside-box mask, and recombined with the outside bounding box mask. The method is visualized in Figure 4.

Formalized, mask M distinguishes between the inside and outside of bounding boxes. Let I be represent an original training image, and A the phase augmentation function, which could be different for inside (A_{in}) and outside (A_{out}) of the bounding boxes.

$$M(x,y) = \begin{cases} 1 & \text{if } (x,y) \text{ is inside any bounding box} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

The augmented image I' can then roughly be defined as:

$$\begin{aligned} I'(x,y) = \quad & M(x,y) \odot A_{\text{in}}(x,y) + \\ & (1 - M(x,y)) \odot A_{\text{out}}(x,y) \end{aligned} \tag{2}$$

Where x and y are the pixel coordinates, and $\odot$ denotes element-wise multiplication. The FFT computation in A_{in} is computed only once on an area from $(\min(x), \min(y))$ to $(\max(x), \max(y))$.

The objectives are twofold: to greatly reduce the area where the FFT operation needs to be computed, thereby reducing memory usage and improving speed, and to make the objects stand out more prominently compared to the rest of the training image. However, this idea introduces additional complexities and obstacles. When reblending the images, mismatched edges may occur where the two masks meet, potentially confusing the model during training. Furthermore, there is a risk that the model may overfit to the artificial boundaries created by the masks, learning to recognize these boundaries rather than generalizing from actual object features. To solve this, a feathering technique is applied at the boundaries between the inside and outside masks to ensure a smooth transition that appears more natural to the model. A radial feathering technique is used, from the center of the bounding box letting through the most, to edges the least and the most of the original image is visible.

4. Experimental design

4.1. Datasets

The datasets used for training, validation, and testing are two of the most popular benchmark datasets for object detection: PASCAL VOC and Microsoft COCO, see Table 1 for details. **COCO 2017** (Common Objects in Context) [13] features diverse images with detailed annotations for object detection, segmentation, and captioning. The training and validation sets include over 120,000 images and 80 object classes. **PASCAL VOC 2007 and 2012** [4] provide a wide range of images across different scenes, featuring common objects in varied poses and occlusions. They cover 20 object classes, including vehicles, household items, and animals. The datasets include over 16,000 images containing around 30,000 ROI-annotated objects. The combined dataset is much smaller than COCO and has far fewer objects, and is therefore used for the smaller-scale experiments. Furthermore, the dataset is more evenly split between training and validation images, whereas COCO 2017 has a training-to-validation ratio of approximately 24:1.

In order to evaluate performance on corrupted data, the validation sets of both aforementioned datasets were used to generate **COCO-C** and **Pascal VOC-C** (2007). These corrupted datasets are generated using an image corruption library referenced in [14], which applied 15 types of corruption at 5 increasing severity levels. The corruptions include various forms of noise, blur, weather-related effects, lighting changes, and compression artifacts, simulating real-world conditions to test model robustness.

4.2. Preprocessing

For all data augmentation methods a consistent image resolution is required. Since these techniques are computationally intensive, the chosen resolution should balance between computational cost and model performance, i.e. the images should not be too large. Ultimately, a 224×224 resolution is chosen. This is a common resolution in computer vision tasks, which also has been used in the studies of APR and VIPAug.

4.3. Evaluation metrics

Evaluating the effectiveness of the different augmentation methods requires a comprehensive set of metrics that cover all aspects, and accurately reflect the performance of the models in terms of detection accuracy, localization precision, and computational efficiency. This section outlines the metrics that are used to assess the object detection models trained with and without the applied phase data augmentation techniques.

The **mean average precision (mAP)** is the primary metric for evaluating the performance of object detection models. It calculates the average precision (AP) across all categories by taking the mean of the APs for each class.

The **intersection over union (IoU)** assesses object localization precision by measuring the overlap between predicted and actual bounding boxes.

The **mean performance under corruption (mPC)** is used to assess benchmark performance on corrupted data:

$$\mathrm{mPC} = \frac{1}{N_c} \sum_{c=1}^{N_c} \frac{1}{N_s} \sum_{s=1}^{N_s} P_{c,s} \tag{3}$$

Where N_c represents the number of corruptions and N_s the number of severity levels of corruptions [14].

Furthermore, the **relative performance under corruption (rPC)** measures the how much the performance gets worse under corruptions as opposed to under a clean performance, noted as P_{clean} [14]. The rPC is defined as follows:

$$\mathrm{rPC} = \frac{\mathrm{mPC}}{P_{\text{clean}}} \tag{4}$$

4.4. Full dataset experiments

For the experiments on both the benchmark datasets, these are the models used for the:

- The **YOLOv8** model is the pre-trained `YOLOv8n.pt`, configured with a learning rate of 0.002 and cosine learning rate scheduling over 250 epochs, with a batch size of 64.
- The **RT-DETR** model was trained using `rtdetr-l.yaml`, with a learning rate of 0.001, for 250 epochs. While most models were trained with a batch size of 64, certain experiments were conducted with a reduced batch size of 32 to address memory constraints. This difference in batch size is not expected to significantly impact the comparability of results.

Both models have a patience, also known as early stopping value, of 100 epochs to prevent overfitting when results are not improving. For reproducibility a fixed seed of 0 is used.

No augmentation and default augmentation baselines. The baseline experiments established the standard performance of object detection models without any data augmentation. This provides a reference point to assess the effectiveness of different augmentation methods. Additionally, for both models a default augmentation baseline was established using the standard data augmentation pipelines that belongs to each model for comparison.

APR-S and VIPAug-G standalone. For these experiments, APR-S and VIPAug-G were tested as standalone augmentation methods. All default augmentations are disabled, only the *LetterBox* transformation

Table 1. Comparison of the benchmark datasets.

Dataset	Classes	Number of images			Number of objects		
		Training	Validation	Total	Training	Validation	Total
Pascal VOC 2007	20	2,501	2,510	5,011	6,301	6,307	12,608
Pascal VOC 2012	20	5,717	5,823	11,540	13,609	13,841	27,450
COCO 2017	80	118,287	5,000	123,287	860,001	36,781	896,782

was applied beforehand to ensure consistent image sizing across the dataset. To implement this, YOLOv8 and RT-DETR were modified by extending their respective `DetectionTrainer` classes. Specifically by adding a custom `YOLODataset` class that allows for the phase data augmentation to happen at the end of the pipeline. The code of APR and VIPAug were reused, and modified so that they take a label class instead of a single image. This code can be reused for any object detection model. The standard deviations of the Gaussian distributions for VIPAug-G are set to $\sigma_{\text{vital}} = 0.001$ and $\sigma_{\text{nonvital}} = 0.014$. This setup allowed the evaluation of how these phase data augmentation methods perform without the influence of the default traditional augmentations, providing insight into their isolated effectiveness. Initial tests were conducted with and without bounding box correction. As results showed improved performance with bounding box correction, it was applied in all subsequent experiments.

Combining default and phase data augmentation. Additionally, APR-S and VIPAug-G were placed at the end of the default augmentation pipeline, so that their contributions could be assessed in combination with traditional visual augmentations. This approach aimed to understand the added value these methods brought to model robustness and generalization when integrated into a more comprehensive augmentation strategy. The reason why they are put at the end, besides that is also how it is implemented in classification, is to ensure that phase data augmentation is the final step shaping the input image. The outcomes of all these experiments provide the answer to sub-questions 1 and 2.

Regional phase data augmentation experiments. The regional phase data augmentation method as described in Section 3.2 was tested by picking the best performing phase data augmentation method, and apply it inside bounding boxes only. The previous experiments serve as a baseline to be compared with.

In smaller scale experiments, multiple different intensities of recombination are applied inside the bounding boxes, also varying different feathering techniques. to assess their impact on model performance. Model performance will be evaluated using the same metrics as the previous experiments. The outcomes of these experiments provide the answer to sub-question 3.

4.5. Robustness experiments

To evaluate the robustness of the trained models described above, the COCO-C dataset was used. Each model was validated across all corruption types at five severity levels. For each corruption the same label file was used for the labels of the images. The experimental pipeline involved configuring datasets, copying label files, and running evaluations. Metrics such as mAP and IoU were recorded in CSV format for analysis. These experiments provided an insight into how the addition of phase data augmentations impacts the robustness of object detection models when exposed to image corruptions.

4.6. Smaller scale experiments

To ensure a practical training process within a reasonable time frame, the main experiments were conducted using a relatively small image size of 224×224 and YOLOv8n, which is the smallest model in YOLOv8. However, relying solely on results from these constrained settings may present a distorted view of the impact of phase data augmentation. For instance, the augmentation may show a positive effect primarily because the baseline model is relatively weak, while its benefits might diminish, or even reverse, when applied to larger, more capable models. To further explore how the image size and model size might have impacted these results, additional experiments were run multiple times on the smaller VOC dataset, allowing for faster testing with larger image sizes and more computationally demanding models. To ensure reproducibility, fixed seeds ranging from 0 to n (where n represents the number of experiment repetitions) were used.

5. Results

5.1. Performance on standard datasets

Table 2 compares the different methods of augmentation on two models. The input size has been set to 224×224 for all models, to ensure fair comparison and not too long computation times. For some RT-DETR models, the batch size had to be reduced to 32 or even 16 for some models

Table 2. Results for YOLOv8n and RT-DETR-l across different augmentation methods on the COCO and VOC datasets. The models were trained on COCO *train2017* and Pascal VOC *train2007* and *train2012*, and validated on COCO *val2017* and Pascal VOC *val2007*. "#Ep. (best)" indicates the total training epochs, with the best-performing epoch with regards to mAP_{50-95} in parentheses. The highest value of a column is highlighted in bold.

	Augmentation method	COCO 2017 #Ep. (best)	Precision	Recall	mAP_{50}	mAP_{50-95}	Pascal VOC #Ep. (best)	Precision	Recall	mAP_{50}	mAP_{50-95}
YOLOv8n											
1.	None	217 (117)	0.505	0.293	0.299	0.189	196 (96)	0.568	0.417	0.441	0.276
2.	Default	250 (250)	0.529	**0.325**	0.338	0.221	250 (197)	0.697	0.589	0.643	0.451
3.	Only APR-S	250 (200)	0.500	0.301	0.313	0.201	174 (74)	0.577	0.453	0.477	0.296
4.	Only VIPAug-G	250 (163)	0.495	0.303	0.311	0.201	178 (78)	0.600	0.449	0.477	0.293
5.	Default + APR-S	250 (250)	0.510	0.322	0.333	0.218	250 (250)	**0.706**	0.579	0.644	0.456
6.	Default + VIPAug-G	250 (250)	0.519	0.319	0.332	0.216	250 (250)	0.698	0.584	0.648	0.456
7.	BBox APR-S	250 (140)	0.495	0.299	0.307	0.195	223 (123)	0.605	0.457	0.481	0.302
8.	BBox VIPAug-G	221 (121)	0.505	0.298	0.307	0.196	228 (128)	0.595	0.443	0.473	0.297
9.	Default + BBox APR-S	250 (250)	**0.532**	0.324	**0.339**	**0.222**	250 (231)	0.696	**0.604**	**0.653**	**0.458**
10.	Default + BBox VIPAug-G	250 (250)	0.527	0.320	0.339	0.222	250 (248)	0.683	0.597	0.650	0.457
RT-DETR-l											
1.	None	250 (98)	0.537	0.393	0.398	0.261	139 (39)	0.332	0.237	0.198	0.115
2.	Default	250 (239)	0.674	**0.485**	**0.524**	**0.362**	250 (234)	0.728	0.606	0.658	0.466
3.	Only APR-S	250 (177)	0.609	0.436	0.451	0.304	222 (122)	0.554	0.371	0.383	0.239
4.	Only VIPAug-G	250 (181)	0.596	0.432	0.448	0.302	250 (152)	0.541	0.363	0.370	0.234
5.	Default + APR-S	250 (249)	**0.675**	0.473	0.515	0.356	250 (250)	0.733	0.607	**0.673**	**0.484**
6.	Default + VIPAug-G	250 (250)	0.668	0.472	0.512	0.353	250 (250)	0.720	0.610	0.664	0.477
7.	BBox APR-S	250 (132)	0.591	0.426	0.438	0.293	242 (142)	0.518	0.341	0.352	0.215
8.	BBox VIPAug-G	250 (172)	0.585	0.434	0.445	0.297	250 (194)	0.523	0.345	0.358	0.223
9.	Default + BBox APR-S	250 (250)	0.666	0.482	0.520	0.358	250 (250)	0.736	**0.613**	0.669	0.479
10.	Default + BBox VIPAug-G	250 (250)	0.669	0.482	0.521	0.360	250 (244)	**0.745**	0.582	0.655	0.468

due to memory constraints. For both models, on the Pascal VOC dataset, higher performance over all metrics was achieved using a combination of the default augmentation and phase data augmentation. When the proposed augmentation method was added to the default pipeline, a 1 percentage point improvement is observed. On the larger COCO 2017 dataset, the results were more similar when using the default augmentation, showing no notable improvements by adding phase data augmentation. Without the default augmentation, the models trained with only phase data augmentation were at least still better than no augmentation, but did not beat the default augmentation method.

These results demonstrate that the phase data augmentation methods can effectively complement traditional augmentations, improving overall model performance slightly, particularly on smaller datasets, but at the cost of longer training times. It is also clear that in no way phase data augmentation is sufficient on their own.

5.2. Robustness under corrupted conditions

The models from Table 2 were validated using the COCO-C and Pascal VOC-C datasets to assess their robustness under various corrupted conditions. Table 3 summarizes the mPC and rPC values (Eq. 3 and 4 respectively) for all corruption types. In general, the models trained with both default and phase data augmentations achieved

Table 3. Robustness of YOLOv8 and RT-DETR models trained with several augmentation methods on COCO-C and Pascal VOC-C. The augmentation methods 1 to 10 correspond to the respective order of Table 2.

Augmentation method	COCO-C clean P	COCO-C corrupted mPC	COCO-C relative rPC	Pascal VOC-C clean P	Pascal VOC-C corrupted mPC	Pascal VOC-C relative rPC
YOLOv8n						
1	0.505	0.378	0.749	0.568	0.400	0.704
2	0.529	0.414	0.783	0.697	0.537	0.771
3	0.500	0.412	0.824	0.577	0.457	0.792
4	0.495	0.406	0.821	0.600	0.470	0.784
5	0.510	**0.435**	**0.854**	**0.706**	0.589	0.834
6	0.519	0.433	0.834	0.698	**0.590**	**0.845**
7	0.495	0.396	0.800	0.605	0.466	0.771
8	0.505	0.391	0.774	0.595	0.454	0.763
9	**0.532**	0.428	0.805	0.696	0.569	0.818
10	0.527	0.429	0.814	0.683	0.572	0.837
RT-DETR-l						
1	0.537	0.450	0.838	0.332	0.240	0.724
2	0.674	0.532	0.790	0.728	0.554	0.760
3	0.609	0.518	0.851	0.554	0.428	0.772
4	0.596	0.519	**0.871**	0.541	0.410	0.757
5	**0.675**	**0.573**	**0.849**	0.733	**0.624**	0.852
6	0.668	**0.573**	0.858	0.720	0.617	**0.857**
7	0.591	0.491	0.831	0.518	0.372	0.717
8	0.585	0.498	0.851	0.523	0.371	0.709
9	0.666	0.554	0.832	0.736	0.583	0.793
10	0.669	0.558	0.834	**0.745**	0.576	0.773

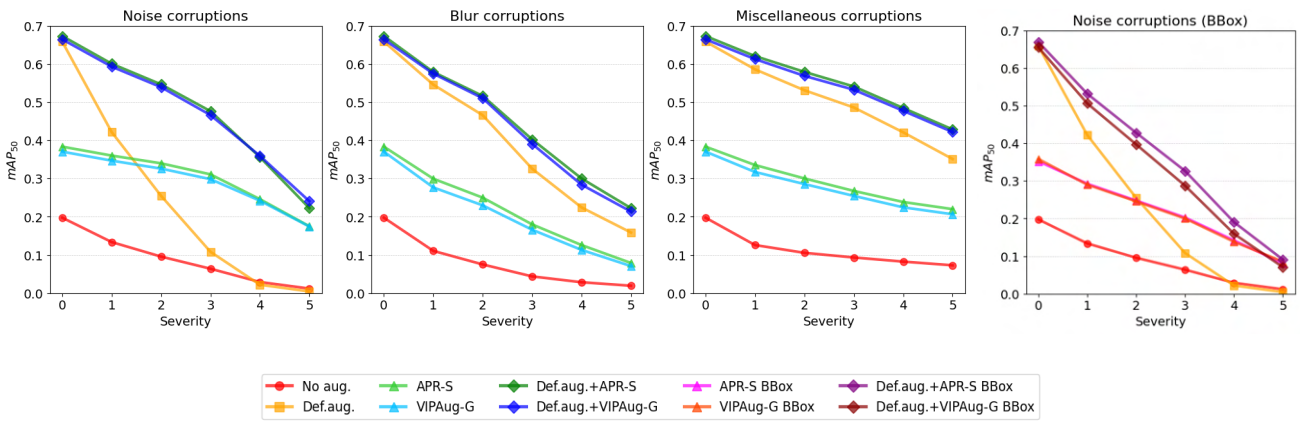


Figure 5. Aggregated mean mAP_{50} for ten RT-DETR models trained with different data augmentation methods, validated on Pascal VOC-C. The corruptions are grouped by noise, blur, and miscellaneous corruptions across five severities of increasing intensity. Phase data augmentation methods show a significantly less steep drop in performance on noise corruptions as the severity increases, compared to the default augmentation method. However, the BBox-specific phase data augmentation method (rightmost) is less robust to noise corruptions than the standard phase data augmentation methods.

the highest precision and also were the most robust. To further analyze trends, Figure 5 groups corruptions into noise, blur, and miscellaneous corruption categories, providing a more detailed comparison of performance drops across severity levels. As shown, the performance drop for phase-augmented models is less steep under noise corruptions, demonstrating their effectiveness in maintaining robustness. Models with phase data augmentations consistently achieved higher precision scores for noise corruptions compared to default augmentations, indicating improved robustness to high-frequency distortions. However, some of these models performed worse under some blur corruptions, with lower precision scores than their counterparts using only default augmentations.

5.3. Smaller scale experiments

The smaller-scale experiments were conducted to examine the effects of image size and model size on performance. These experiments were performed five times, and the results are averaged.

5.3.1 Impact of image size

Due to impact on training time of having a large image size, the main experiments were performed with an image size of 224×224. However, additional tests with a larger image size of 640×640 were conducted to assess the effect of resolution on model performance. As shown in Table 4, increasing the image size consistently improved mAP metrics across both default and APR-S configurations. For the smaller image size, the relative improvement of APR-S compared to the default pipeline was 0.3% for mAP_{50} and 0.7% for $mAP_{50\text{-}95}$. In contrast, with the larger image size, the relative improvement of APR-S increased to 0.6% for mAP_{50} and 1.1% for $mAP_{50\text{-}95}$. This suggests that the benefits of APR-S scale slightly more effectively with higher resolution inputs, possibly due to the additional structural detail present in larger images, which allows the model to capture finer details, which in turn allows the phase data augmentation to exploit this, but at the cost of a logarithmically increased training time.

Table 4. Averaged performance with different image sizes over five runs.

Image size	Default		Default + APR-S	
	mAP_{50}	$mAP_{50\text{-}95}$	mAP_{50}	$mAP_{50\text{-}95}$
224	0.645	0.452	0.647	0.455
640	0.789	0.582	0.794	0.589

5.3.2 Impact of model size

The smaller dataset allowed for experimentation with larger versions of YOLOv8, namely YOLOv8s and YOLOv8m, see Table 5. Interestingly, YOLOv8m with APR-S underperformed compared to the default version in terms of overall mAP metrics, despite achieving higher precision during training. Specifically, the precision with APR-S reached 0.804, surpassing the default version's maximum precision of 0.792. This suggests that while APR-S may enhance certain aspects of detection, it may not consistently improve overall performance across all metrics, particularly with larger models.

Table 5. Averaged performance of varying model sizes over five runs.

YOLO model	#Param. (M)	Default		Default + APR-S	
		mAP_{50}	$mAP_{50\text{-}95}$	mAP_{50}	$mAP_{50\text{-}95}$
v8n	3.2	0.645	0.452	0.647	0.455
v8s	11.2	0.715	0.516	0.726	0.528
v8m	25.9	0.781	0.586	0.763	0.569

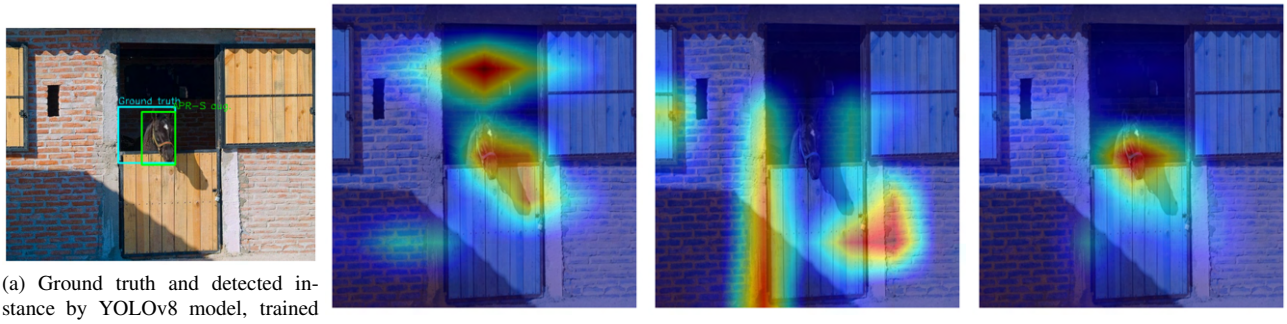
6. Discussion

The **key findings** of this research are that phase data augmentations, such as APR-S and VIPAug-G, complement traditional augmentation methods in object detection. Not only by slightly enhancing mAP, but also improving the robustness of object detection models, particularly under noisy conditions. This highlights the potential of using augmentations in the phase domain to refine feature representations, especially for detection tasks in challenging scenarios like noisy or weather-impacted environments. In general, models trained with these methods demonstrated consistent improvements in $mAP_{50\text{-}95}$compared to those trained with only default augmentations, with gains of around 1 to 2 percentage point observed in specific configurations.

The **error analysis** thereafter was inconclusive, as no clear pattern could be identified to explain where the phase data augmentation outperformed the default model. In some images, the default model performed better, while in others, the phase-augmented model was better even when the image content appeared to be similar. On average, the phase-augmented model showed slightly better results, but the absence of a consistent trend makes it difficult to draw definitive conclusions. This is exemplified in Figure 6, where the model with arguably the worst class activation is the only one that detects the correct instance. Additionally, as shown in Table 2, phase data augmentation led to a decrease in recall for most models. This suggests that while phase data augmentation improves precision, it does not necessarily validate the idea that it enhances detection by diversifying shape variations. Instead, it appears to make the model more conservative, focusing on avoiding false positives at the expense of missing more objects. For instance, on the VOC 2007 dataset with YOLOv8n, the number of detected instances decreased from 4,128 with default augmentations to 3,707 with *Default + APR-S*, and 3,820 with *Default + BBox APR-S*. Similarly, the average confidence scores followed a similar pattern, dropping from 0.782 for the default model to 0.772 for *Default + APR-S*, and further to 0.764 for *Default + BBox APR-S*. These findings indicate that phase data augmentation may over-regularize the model, reducing its overall confidence and recall, instead of allowing it to detect difficult-to-detect objects. However, this might also be caused by the specific implementation and the chosen hyperparameters used in this research.

The **BBox phase data augmentation** results are consistent with prior work that demonstrated applying standard augmentations specifically within bounding boxes can yield slight performance benefits [2]. This underscores the potential of targeted augmentations tailored to specific object classes, which could be further refined by dynamically adjusting augmentation parameters based on the objects themselves, such as applying different intensities to objects that the model does not detect well. Using BBox-only augmentation is also promising from a computational perspective, as it requires performing FFT operations on smaller area rather than the entire image, thereby reducing computational overhead. However, the current approach has notable limitations. First of all, the method performs worse than the standard phase data augmentation in terms of corruption robustness. This could be because the reduced area of the FFT operation and the feathering with the original image introduce less variety in the augmented data, limiting the ability to generalize to corruptions such as noise and blur. Secondly, the radial feathering technique does not account for the distribution of features within the bounding box and is applied uniformly from the center outward, potentially missing critical details in non-central areas, where the shape of objects are more likely to be defined. Besides that, the current policy of deciding the part that gets augmented is overly simplistic and can lead to inefficiencies, particularly in the case of images with bounding boxes that are located far apart from each other, near the edges of the image. In such cases, the extracted area still encompasses nearly the entire image, undermining the goal of reducing the area of the FFT operation. A clustered approach, grouping closely positioned bounding boxes and calculating the FFT per cluster, could address this issue and further improve the efficiency of this approach. This approach would also handle images with overlapping bounding boxes more effectively. In the current implementation, feathering was applied sequentially to each individual bounding box, but applying it per cluster would be both computationally more efficient and provide smoother transitions.

Regarding **robustness**, the improved performance of phase-augmented models under noise corruption scenarios is promising, suggesting that phase data augmentation enhances robustness to high-frequency distortions. However, its practical utility in real-world applications remains uncertain. Namely, if robustness to noise were to be a primary requirement, explicitly training models with noise augmentations might be a more direct and effective approach. It is also possible that the observed robustness to noise is solely due to the similarity between the effects of phase data augmentation and random noise corruption, rather than an inherent improvement in model robustness. Besides that, phase data augmentations showed a slight but notable drop



(a) Ground truth and detected instance by YOLOv8 model, trained with *Default* + *APR-S* augmentation.

(b) Default augmentation.

(c) Default + APR-S.

(d) Default + BBox APR-S.

Figure 6. Class activation mappings (CAM) on an example validation image using three differently trained YOLOv8 models. This figure illustrates how each method can result in completely different areas of focus for detection.

in performance under blur corruptions in some cases, indicating a limitation in addressing low-frequency distortions.

Optimal configurations for integrating phase data augmentations remain a critical area for further investigation. After all, some of the aforementioned limitations could partly be attributed to suboptimal configurations, such as the chosen hyperparameters or phase data augmentation settings. Results indicate that factors such as image size and model size influence the impact of phase data augmentation on performance, which in turn indicates that phase data augmentation might be more suitable on simpler models with less training data. Also, this research did not explore combinations of phase data augmentation with other augmentation methods, such as mixup, which could potentially provide even more complementary improvements.

Limitations and challenges that remain include the substantial memory overhead and longer training times introduced by Fourier transforms. With a computational complexity of $O(n \log n)$, FFT operations can hinder scalability, particularly for large datasets. Furthermore, for a fairer comparison between models, when the batch size is reduced, the learning rate could have been scaled proportionally, using either linear scaling with $\sqrt{k}$ or k [5, 9]. However, since an Adam optimizer was used, which adapts the learning rate dynamically, this issue was partially mitigated. Moreover, bounding box-specific augmentations introduced potential artifacts, such as mismatched edges between augmented regions. Feathering techniques helped mitigate these issues, but further refinements are necessary to ensure seamless integration and avoid confusing the model.

Potential refinements and future work include conducting a comprehensive sensitivity analysis to refine configuration hyperparameters, particularly for bounding box-specific augmentations. Then, improvements in blending techniques and combining with different augmentation methods could further improve the practical utility and scalability of (BBox) phase data augmentation methods.

7. Conclusion

This research investigated the applicability and effectiveness of phase data augmentation for object detection models. By implementing and testing APR-S and VIPAug-G on popular datasets COCO and Pascal VOC, it was demonstrated that phase data augmentation methods can enhance detection quality, particularly under noisy conditions. The experiments showed that integrating phase data augmentation led to slight improvements in precision and mAP but sometimes at the expense of recall. To address the supporting research questions:

How do current phase data augmentation methods, as used in image classification, affect the performance of object detection models compared to visual data augmentation methods? Phase data augmentation methods, such as APR-S and VIPAug-G, showed modest but consistent improvements in mAP when applied to object detection models, similar to the modest accuracy gains observed in image classification tasks. However, phase data augmentations alone were not sufficient to achieve significant performance gains, as their effectiveness was only apparent when combined with visual augmentations.

Among existing phase and visual data augmentation methods, which one, or what combination thereof, maximizes the performance of object detection models? The combination of default visual augmentations with APR-S or VIPAug-G mostly yielded the best performance. Phase data augmentations alone improved compared to no augmentation, but were most effective when integrated with traditional visual augmentations, highlighting the importance of a hybrid approach. Default augmentations added more spatial and color variability, which in turn was amplified by the phase data augmentation. The most effective configuration was *Default* + *APR-S*, which provided slight but consistent improvements across multiple metrics and datasets, particularly for smaller-scale experiments, and the best overall robustness.

Are there potential refinements (such as selective augmentation) to current phase data augmentation methods that would improve the performance of object detection models? Several potential refinements were identified. The BBox-specific augmentation strategy demonstrated potential by targeting phase augmentations within bounding boxes, reducing computational overhead and training time. However, the current implementation was found to be too simplistic, and more sophisticated approaches, such as cluster-based or object-aware strategies, could further enhance performance by tailoring augmentations to challenging instances or regions of interest.

The overarching research question posed by this research was: ***How can the performance of varying object detection models be enhanced by using phase data augmentation methods?*** This research demonstrated that integrating phase data augmentation methods into the augmentation pipeline can enhance the performance and robustness of object detection models, particularly with limited model size and or training data, provided that it is combined with traditional augmentations. While the improvements were relatively modest, they highlight the complementary nature of phase augmentations in enhancing structural representations, especially when exposed to noise corruptions. Furthermore, the research identified specific areas for refinement, such as targeted application within bounding boxes, that could unlock further potential for phase data augmentations in object detection tasks.

References

[1] Ayoub Benali Amjoud and Mustapha Amrouch. Object Detection Using Deep Learning, CNNs and Vision Transformers: A Review. *IEEE Access*, 11:35479–35516, 2023. 2

[2] Ruichu Cai, Zijian Li, Pengfei Wei, Jie Qiao, Kun Zhang, and Zhifeng Hao. Learning Disentangled Semantic Representation for Domain Adaptation. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, pages 2060–2066, Aug. 2019. arXiv:2012.11807 [cs]. 10

[3] Guangyao Chen, Peixi Peng, Li Ma, Jia Li, Lin Du, and Yonghong Tian. Amplitude-Phase Recombination: Rethinking Robustness of Convolutional Neural Networks in Frequency Domain, Aug. 2021. arXiv:2108.08487 [cs]. 1, 3, 4

[4] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. 6

[5] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour, Apr. 2018. arXiv:1706.02677 [cs]. 11

[6] Dan Hendrycks, Norman Mu, Ekin D. Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. AugMix: A Simple Data Processing Method to Improve Robustness and Uncertainty, Feb. 2020. arXiv:1912.02781 [cs, stat]. 3

[7] Glenn Jocher, Ayush Chaurasia, and Jing Qiu. Ultralytics YOLOv8, 2023. 2

[8] Parvinder Kaur, Baljit Singh Khehra, and Er. Bhupinder Singh Mavi. Data Augmentation for Object Detection: A Review. In *2021 IEEE International Midwest Symposium on Circuits and Systems (MWSCAS)*, pages 537–543, Lansing, MI, USA, Aug. 2021. IEEE. 2

[9] Alex Krizhevsky. One weird trick for parallelizing convolutional neural networks, Apr. 2014. arXiv:1404.5997 [cs]. 11

[10] Debasis Kumar and Naveed Muhammad. Object Detection in Adverse Weather for Autonomous Driving through Data Merging and YOLOv8. *Sensors*, 23(20):8471, Oct. 2023. 3

[11] Ingyun Lee, Wooju Lee, and Hyun Myung. Domain Generalization with Vital Phase Augmentation, Jan. 2024. arXiv:2312.16451 [cs]. 1, 3, 4

[12] Jinlong Li, Runsheng Xu, Jin Ma, Qin Zou, Jiaqi Ma, and Hongkai Yu. Domain Adaptive Object Detection for Autonomous Driving under Foggy Weather. In *2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 612–622, Waikoloa, HI, USA, Jan. 2023. IEEE. 3

[13] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft COCO: Common Objects in Context, Feb. 2015. arXiv:1405.0312 [cs]. 6

[14] Claudio Michaelis, Benjamin Mitzkus, Robert Geirhos, Evgenia Rusak, Oliver Bringmann, Alexander S. Ecker, Matthias Bethge, and Wieland Brendel. Benchmarking Robustness in Object Detection: Autonomous Driving when Winter is Coming, Mar. 2020. arXiv:1907.07484 [cs, stat]. 3, 6

[15] Alan V. Oppenheim and Jae Sung Lim. The importance of phase in signals. *Proceedings of the IEEE*, 69:529–541, 1980. 1

[16] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6):1137–1149, June 2017. 2

[17] F. Sultana, A. Sufian, and P. Dutta. A Review of Object Detection Models based on Convolutional Neural Network. volume 1157, pages 1–16. 2020. arXiv:1905.01614 [cs]. 2

[18] Juan Terven and Diana Cordova-Esparza. A Comprehensive Review of YOLO Architectures in Computer Vision: From YOLOv1 to YOLOv8 and YOLO-NAS. *Machine Learning and Knowledge Extraction*, 5(4):1680–1716, Nov. 2023. arXiv:2304.00501 [cs]. 2

[19] Qinwei Xu, Ruipeng Zhang, Ya Zhang, Yanfeng Wang, and Qi Tian. A Fourier-based Framework for Domain Generalization, May 2021. arXiv:2105.11120 [cs]. 1, 3

[20] Mehmet Kerim Yucel, Ramazan Gokberk Cinbis, and Pinar Duygulu. HybridAugment++: Unified Frequency Spectra

Perturbations for Model Robustness. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5695–5705, Paris, France, Oct. 2023. IEEE. 1

[21] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. CutMix: Regularization Strategy to Train Strong Classifiers with Localizable Features, Aug. 2019. arXiv:1905.04899 [cs]. 3

[22] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond Empirical Risk Minimization, Apr. 2018. arXiv:1710.09412 [cs]. 3

[23] Yian Zhao, Wenyu Lv, Shangliang Xu, Jinman Wei, Guanzhong Wang, Qingqing Dang, Yi Liu, and Jie Chen. DETRs Beat YOLOs on Real-time Object Detection, Apr. 2024. arXiv:2304.08069 [cs]. 2

[24] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random Erasing Data Augmentation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(07):13001–13008, Apr. 2020. 3

A. Corruption examples

(a) Gaussian noise.

(b) Motion blur.

(c) Snow.

Figure 7. Examples of a noise, a blur, and a miscellaneous corruption at severity levels 1 and 5, respectively.

B. BBox correction policy

Table 6. Both APR-S and VIPAug-G make use of two transformation in their augmentation process. The resulting image is a recombination of the amplitude (A) and phase (P) of both transformed images. This table indicates when bounding box correction should be applied.

Recombination	A**(1st aug.),** P**(2nd aug.)**	A**(2nd aug,),** P**(1st aug.)**
Both non-spatial **(e.g. solarize and equalize)**	None	None
1st non-spatial, then 2nd spatial **(e.g. solarize and translate-y)**	2nd aug.	None
1st spatial, then 2nd non-spatial **(e.g. translate-y and solarize)**	1st aug.	1st aug.
Both spatial **(e.g. translate-y, translate-x)**	Both	Both

C. Robustness results per corruption

Table 7. $mAP_{50\text{-}95}$ for each corruption on COCO-C, averaged over all severities, using YOLOv8n.

Corruption	**mAP50-95**		
	Def. aug.	**Def. aug. + APR-S**	**Relative**
Impulse noise	0.077	0.130	68.48%
Gaussian noise	0.085	0.128	50.13%
Shot noise	0.091	0.134	47.24%
Defocus blur	0.162	0.185	14.13%
Glass blur	0.144	0.163	13.09%
Snow	0.099	0.111	11.95%
Zoom blur	0.062	0.067	9.36%
Frost	0.130	0.140	7.98%
JPEG compression	0.190	0.203	6.75%
Motion blur	0.140	0.148	5.89%
Elastic transform	0.197	0.198	0.48%
Brightness	0.199	0.200	0.48%
Pixelate	0.220	0.217	-1.37%
Fog	0.175	0.167	-4.08%
Contrast	0.119	0.110	-6.93%