applied sciences

MDPI

Article

Feature Augmentation Based on Pixel-Wise Attention for Rail Defect Detection

Hongjue Li [1], Hailang Li [2], Zhixiong Hou [2], Haoran Song [2], Junbo Liu [2] and Peng Dai [2,*]

[1] School of Astronautics, Beihang University, Beijing 100191, China
[2] Infrastructure Inspection Research Institute, China Academy of Railway Science Corporation Limited, Beijing 100081, China
* Correspondence: daipeng_iic@protonmail.com

Abstract: Image-based rail defect detection could be conceptually defined as an object detection task in computer vision. However, unlike academic object detection tasks, this practical industrial application suffers from two unique challenges, including object ambiguity and insufficient annotations. To overcome these challenges, we introduce the pixel-wise attention mechanism to fully exploit features of annotated defects, and develop a feature augmentation framework to tackle the defect detection problem. The pixel-wise attention is conducted through a learnable pixel-level similarity between input and support features to obtain augmented features. These augmented features contain co-existing information from input images and multi-class support defects. The final output features are augmented and refined by support features, thus endowing the model to distinguish between ambiguous defect patterns based on insufficient annotated samples. Experiments on the rail defect dataset demonstrate that feature augmentation can help balance the sensitivity and robustness of the model. On our collected dataset with eight defected classes, our algorithm achieves 11.32% higher mAP@.5 compared with original YOLOv5 and 4.27% higher mAP@.5 compared with Faster R-CNN.

Keywords: object detection; pixel-wise attention; feature augmentation; rail defect

Citation: Li, H.; Li, H.; Hou, Z.; Song, H.; Liu, J.; Dai, P. Feature Augmentation Based on Pixel-Wise Attention for Rail Defect Detection. *Appl. Sci.* **2022**, *12*, 8006. https://doi.org/10.3390/app12168006

Academic Editor: Andrés Márquez

Received: 29 June 2022
Accepted: 9 August 2022
Published: 10 August 2022

1. Introduction

Discovering defects on rail is the first step for rail health maintenance and is vital for the safe operation of high speed trains. Recent progress in high-speed photography technology offers the possibility of capturing real-time rail images from a running train and further paved the way to solving this practical industrial problem from the perspective of object detection using computer vision approaches. In computer vision, object detection approaches based on a deep convolutional neural network (CNN) have achieved great progress in both accuracy and efficiency [1–3]. Current CNN-based object detection is mainly built on two alternatives: two-stage methods [4] and one-stage methods [5]. Two-stage methods achieve high detection accuracy by separately conducting region proposal and detection process. As a comparison, one-stage detection methods are less accurate but faster in achieving real-time detection.

In academic research, both two-stage and one-stage methods are usually trained on large-scale benchmarks such as MS COCO [6] and ImageNet [7], and have been successfully applied to various tasks, such as defect detection [8,9], medical detection [10,11], etc. Nevertheless, detecting a defect on the rail image is not as easy as detecting a normal object on natural images due to the following two reasons. First, the concerned defects are usually tiny and ambiguous because the images are captured from a running train at very high speed. Complex illumination environment makes defects look similar to other non-defect patterns such as dirt or gap. Second, there is a lack of benchmark annotations of these railway defects and some defects are difficult to distinguish from 2D images. The insufficient and ambiguous defect images that are substantially different from natural photos may advise against such pretrain-finetune knowledge transfer.

On the basis of increasingly extensive research on attentional feature fusion approaches [12–15], recent detection tasks [16–20] suggested that the idea of exploiting extra features can potentially improve detection performance, especially with insufficient training data. In these attempts, extra information are usually encoded into class-wise/channel-wise feature vectors or pixel-wise relation matrixes for retrieving. However, these methods are proposed for solving the few-shot learning problem. In this work, we borrow the idea of the attentional feature augmentation under the fine-tuning strategy to solve the rail defect detection.

In this paper, we proposed a feature augmentation framework by augmenting input image feature maps with some support feature maps derived from an extra support image set. The input image feature maps are obtained by passing the input image through any backbone neural network which shows no difference than the traditional object detection approach. The novel part of our model are those augmented feature maps that are extracted by developing a pixel-wise metric learning model to generate new feature maps via a query-based attention model. The augmented feature maps bring at least two benefits into the detection framework. First, they alleviate the disturbances of the noise and ambiguities on the original input images. The original input image serves as a query to encourage new feature maps from extra information among those support images. Meanwhile, the pixel-wise attention model can improve the discriminative ability of the generated feature map by imposing the metric learning concept through a series of query-based attention. As a result, these augmented features bring extra information from support images to the final feature maps and hence improve the robustness of the detector for those ambiguous defect patterns. The technical details will be explained in the following section.

2. Related Works

Traditional object detection methods roughly follow the following steps to detect objects: input images, preprocessing, hand-craft features, classification. Many pioneering works mainly focused on the construction of hand-crafted features [21,22] or classification algorithms [23,24]. These methods can achieve good performance in specific types of detection tasks but are less generalizeable to others.

CNN-based object detection methods tried to apply CNN into the previous mentioned detection steps to accelerate the detection speed and improve the generalization ability of the detection model. Earlier attempts separated the detection task into another two steps: region proposal and feature extraction/classification/regression. In these attempts, regions of interest are selected by multiple methods, such as selective search [25], edge box [26], or Region Proposal Network (RPN) [4]. These regions are then sent to a CNN for feature extraction and the subsequent classification and bounding box regression. The above approach is usually called a two-stage method, since it requires two separate steps to achieve detection. Famous two-stage detection methods include Fast RCNN and Faster RCNN. Later CNN-based object detection methods tried to achieve detection through end-to-end training, i.e., to train and detect the image within one step instead of the previously developed two steps. Two typical one-stage detection methods are SSD [27] and YOLO [5] families. To conclude, one-stage detection methods are much faster than two-stage detection methods, but are relatively less accurate.

CNN-based detection methods have been successfully applied to various tasks such as classification [28], detection [29], segmentation [30], etc. In industrial applications, a widely accepted approach is to fine-tune a pretrained network on the scarce samples to achieve task-aware detection ability [8,31,32]. A pretrained model on common large-scale datasets such as ImageNet [7] or COCO [6] can be either fully or partially fine-tuned during implementation. This approach has proved to be effective in various tasks such as defect detection [8,9], medical detection [10,11], etc. However, the insufficient and ambiguous defect images may advise against such pretrain-finetune knowledge transfer.

Many approaches were studied to enhance the detection performance in industrial applications. The image augmentation [33] method is a widely used approach that can

expand the scale of the training dataset through various random changes to the training image. Feature augmentation, on the other hand, is a relatively newer concept and has been used in many tasks such as person re-identification [34] and low-shot learning [35]. Another feature augmentation approach lies in attentional feature fusion [12–15], and has proved to achieve better detection results [16–20] by exploiting extra features. For example, Hu et al. [16] proposed a graph-based Relation R-CNN considering extra global relation in labels and achieved better performance for small objects detection. Yan et al. [19] proposed a predictor-head remodeling network to infer class attentive vectors of low-shot objects, and take channel-wise soft-attention on ROI features. Hu et al. [20] proposed the DCNet with a pretrained ResNet-101 backbone and a pixel-wise dense relation distillation module to aggregate relations between input and support sets. The attentional feature augmentation idea inspired us to implement rail detection with a fine-tuning strategy.

3. Materials and Methods

3.1. Dataset

The rail defect dataset is a series of high resolution (2048 × 2000 pixel of 96 dpi) rail surface images with annotations. We collected 9039 images from the 9 km railway test loop built by the National Academy of Railway Sciences Test Center. The images are taken by CMOS line scan cameras with laser light source and preprocessed by an image processor to eliminate specular reflections. Only 400 of the 9039 images contain objects and are used to build the dataset. The objects in the images can be categorized into four main classes: *damage*, *gap*, *dirt*, and *unknown*. *Damage* class can be further divided into five classes: *general damage* (defects that cannot be categorized into other classes), *dent*, *crush*, *scratch*, and *slant*. A detailed division of the dataset is listed in Table 1.

The dataset is annotated by following the YOLO's annotation format with five numbers. The first number is object type and the following four numbers are object coordinates. Object types are recorded as integers starting from zero, while object coordinates are recorded as four float numbers with six significant digits: x, y, w, and h. x and y are are normalized center coordinates of the bounding box, while w and h are the normalized width and height of the bounding box. The YOLO format can be easily converted to COCO format or PASCA VOC format. Annotated images as examples are illustrated in Figure 1, where blue bounding boxes refer to non-damage features while red boxes refer to damages.

Table 1. Two divisions of the rail defect dataset.

4 Class Division	8 Class Division
gap: gaps left between successive rails on a railway track *dirt*: paint, or mud that covers the surface of the rail *unknown*: unrecognized features	
damage	*general damage*: displacement of parent metal from the rail surface *dent*: tear of the lateral planes of the rail surface *crush*: big/severe wear of the lateral planes of the rail surface *scratch*: small/mild wear of the lateral planes of the rail surface *slant*: tear of the lateral planes of the rail surface

The dataset contains 148 general damages, 180 dirt, 87 unknown, 51 gaps, 35 dent defects, 94 crush defects, 129 scratch defects, and 43 slant defects. The total number of the annotated objects is 767. These annotations are provided by railway maintenance engineers, but still include some mislabeled and unlabeled patterns. We confirmed 11 errors (including wrong labeled and mislabeled objects) within the 767 annotations after double check. The error ratio is approximately 1.43% and is relatively lower than many other widely used datasets [36]. Therefore, we believe the dataset is acceptable for training and detection. Experiments were conducted concerning both 8 and 4 classes (regarding all defect types as one class).

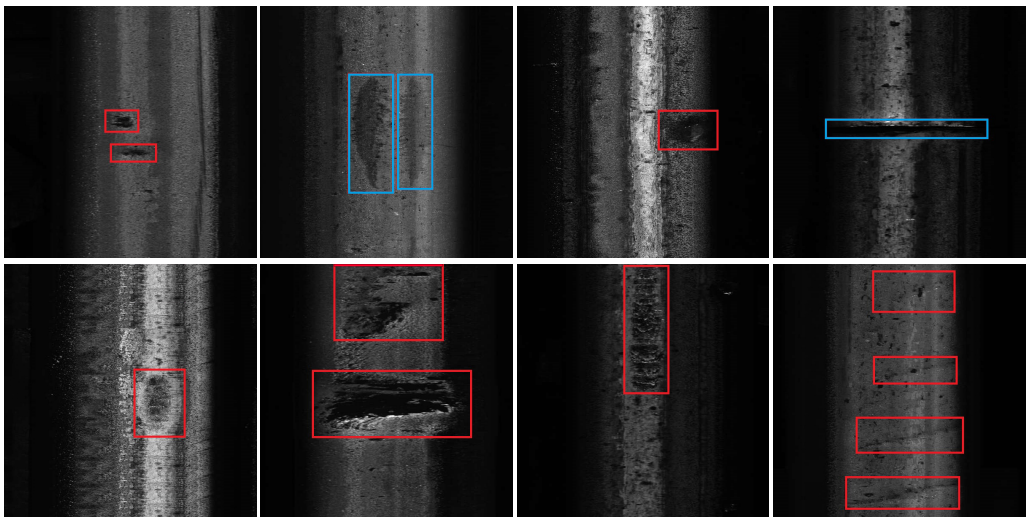


Figure 1. Samples of the rail defect dataset. From left to right, first row contains damage, dirt, unknown and gap, and second row contains dent, crush, scratch and slant.

3.2. Model Architecture

The overall detection model is illustrated in Figure 2. Our model is based on the one-stage object detection architecture to ensure high detection speed. The whole model consists of three parts: (a) a feature extractor as backbone network that builds multi-level semantic feature maps at all scales, (b) the proposed feature augmentation framework based on pixel-wise attention that enables the model to better distinguish ambiguous defect patterns, and (c) multi-level detect heads for both object classification and bounding box regression. To be more specific, the pretrained backbone network is fine-tuned to extract input and support features from an input batch. A novel pixel-level similarity metric is developed to evaluate dense spatial relations that enables pixel-level match of input and support features. The final refined feature are activated by support features for subsequent detector heads.

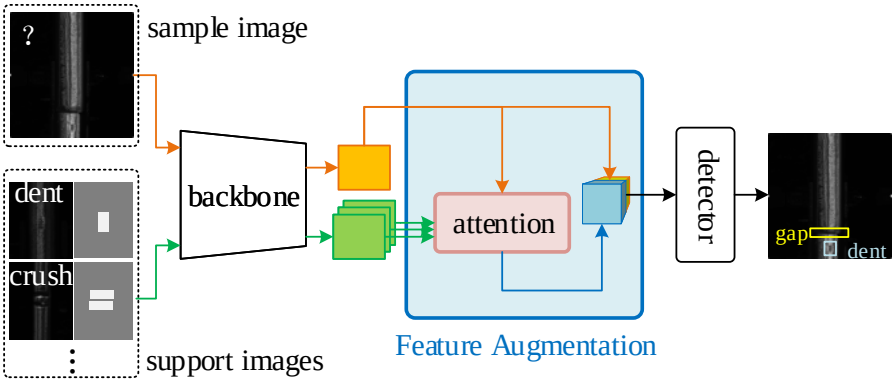


Figure 2. The architecture of the rail defect detection model.

To establish feature augmentation, N numbers of representative defect images are manually selected and masked as support samples. Every single image of the rest of the images is grouped with the N masked images to form a batch of input images. The feature extractor will extract features of the input batch for subsequent feature augmentation.

3.3. Feature Augmentation

To utilize the support defect images and improve class-wise defect detection ability, we perform the feature augmentation framework to aggregate input and support features, as illustrated in Figure 3. The framework mainly consists of a query, key, and value embed-

ding, and a feature fusion operation based on pixel-wise attention. Although there exist alternative embedding and attention methods, we adopt the transformer-style attention [14] because it has been widely used and proved to be generally effective for various tasks.

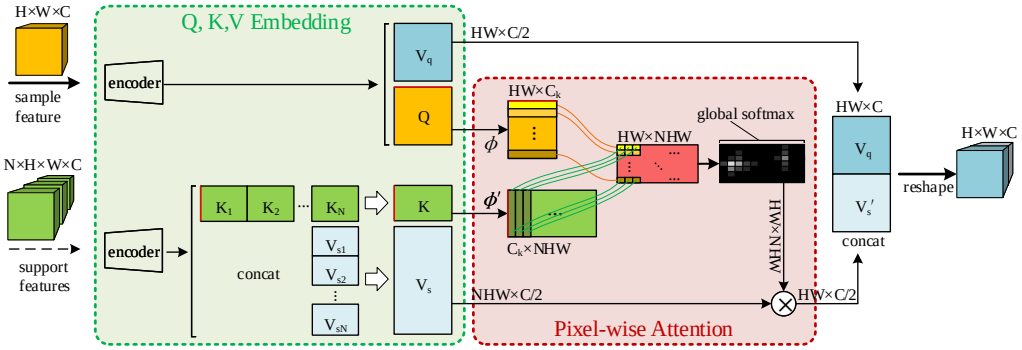


Figure 3. Feature augmentation framework. The sample and support features are first encoded into query (Q), key (K), and value (V) embedding. Two linear transformations are performed on both Q and K feature maps. Pixel-wise attention is performed to fuse the Q and K feature maps to form the final learnable similarity metric, thus retrieving co-existing information from support features.

Query, Key and Value Embedding. The sample and support features extracted by backbone feature extractor are encoded separately to reduce their dimension for saving computation cost. Specifically, sample feature is encoded into a query feature map and a value feature map, while support features are encoded into a concatenated key feature map and a concatenated value feature map. Query and key feature maps contain semantic information about relations, while value maps carry richer detailed information about the original feature maps.

The two encoders share the same structure, i.e., two 3×3 convolution layers, a batch normalization, and a leaky ReLU activation function, but have different weights. The query encoder takes sample feature of size $H \times W \times C$ as input, and output query feature map Q and value feature map V_q as:

$$\begin{aligned} Q &\in \mathbb{R}^{HW \times C_k}, \\ V_q &\in \mathbb{R}^{HW \times C/2}, \end{aligned} \tag{1}$$

where H is feature height, W is feature width, C is channel dimension, and C_k is the user-defined query dimension (e.g., $C_k = C/4$). On the other hand, the support encoder takes N support features as input, and output a series of key feature maps and value feature maps. Support key feature maps are horizontally concatenated, while support value maps are vertically concatenated:

$$\begin{aligned} K &= concat[K_1^T, \cdots, K_N^T] \in \mathbb{R}^{C_k \times NHW}, \\ V_s &= concat[V_{s1}, \cdots, V_{sN}]^T \in \mathbb{R}^{NHW \times C/2}. \end{aligned} \tag{2}$$

Pixel-wise Attention. After acquiring query, key, and value feature maps, pixel-wise attention is performed to activate co-existing patterns among sample and support features. The key of pixel-wise attention lies in a learnable pixel-level similarity metric. To calculate the metric, two different linear transformation ϕ, ϕ' are first applied to query and key feature maps:

$$Q^* = \phi(Q), \quad K^* = \phi'(K). \tag{3}$$

Then, linear transformed query and key feature are fused by inner-production to obtain the pixel-level similarity metric:

$$S = Q^* \cdot K^* = [S^1, S^2, \cdots, S^N] \in \mathbb{R}^{HW \times NHW}. \tag{4}$$

This similarity metric concerning ϕ, ϕ' can be dynamically learned through back-propagation during fine-tuning.

After obtaining the fused similarity metric, global softmax normalization is performed to obtain the final attention weight:

$$\omega = \begin{bmatrix} \omega^1_{1,1} & \cdots & \omega^2_{1,1} & \cdots & \cdots & \omega^N_{1,HW} \\ \vdots & \ddots & \vdots & \ddots & \omega^i_{j,k} & \vdots \\ \omega^1_{HW,1} & \cdots & \omega^2_{HW,1} & \cdots & \cdots & \omega^N_{HW,HW} \end{bmatrix}, \quad \omega^i_{j,k} = \frac{exp(S^i_{j,k})}{\sum_{NHW} exp(S^i_{j,k})}, \tag{5}$$

where $i = 1, 2, \cdots, N$ is the support index, $j, k = 1, 2, \cdots, HW$ is the pixel index of the similarity metric. To be more specific, the rows of the similarity metric correspond to the pixels of the sample feature, while the columns of the metric correspond to pixels of all support features in sequence. The whole metric can be viewed as N blocks $[\omega^1, \omega^2, \cdots, \omega^N]$, each representing relations between the sample feature and one support feature. Global softmax is performed to highlight the most relevant defect patterns while suppress less similar features.

The support value maps can be weighed by ω through another matrix inner-product, and then concatenated with the query value map to form the final output feature map:

$$F = concat[V_q, \omega \cdot V_s] \in \mathbb{R}^{HW \times C}. \tag{6}$$

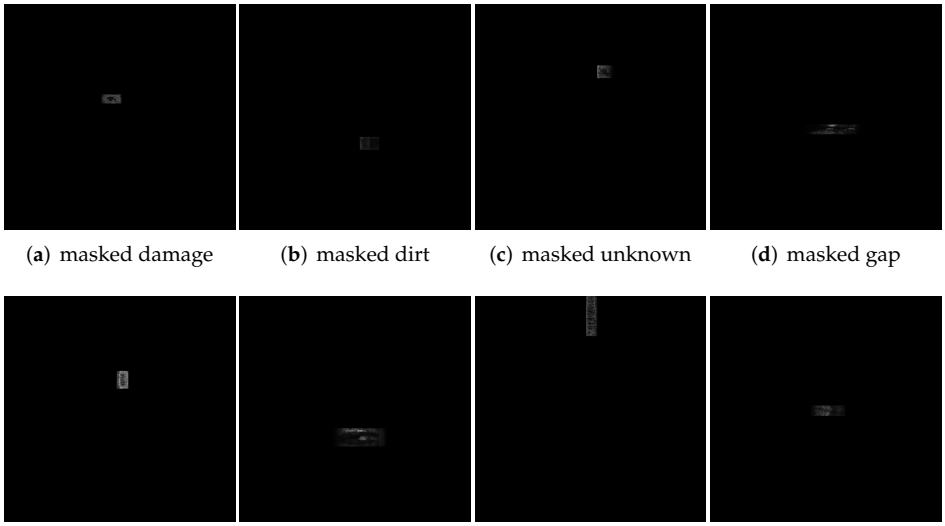
This feature is then reshaped to $H \times W \times C$ to fit the input sample dimension and used for subsequent object classification and bounding box regression.

Our pixel-wise attention within feature augmentation framework can be treated as a combination of self attention and cross attention, since we retrieve hidden representations from both the sample feature and the support features. Previous meta-learning trials either perform class-wise feature vectors to fuse attention feature maps [15] or perform concatenation over N class-wise results to obtain the final soft attention [20]. However, our approach considers a global pixel-wise attention on all N classes, making our model extremely sensitive to pixel-level detailed features of the most relevant support defect, while reducing potential ambiguities from less similar classes or noises. It also makes our model applicable to pretrain-finetune paradigm. In addition, we adopt random selection to generate support class prototype features instead of average calculation to reduce training resource consumption without losing detection performance.

4. Results

In this section, we demonstrate various experimental results to illustrate the effectiveness of our proposed method. All the experiments were conducted on Intel(R) Xeon(R) Gold 6226R CPU@2.90 GHz and NVIDIA RTX 3090 GPU, running an Ubuntu 18.04 operating system. Unless specified otherwise, all used models are implemented based on PyTorch 1.9.1. YOLOv5 and AL-MDN [37] are implemented according to their official GitHub repository, while DyHead [38] and Faster R-CNN [4] are implemented based on Detectron2 [39].

The support defect images are randomly selected from the training set and mask it using its label. One of the chosen result of the eight defect classes are shown in Figure 4. If four classes detection is analyzed, the defect class is randomly chosen from *defect*, *unknown*, *dirt*, and *gap*.



(**a**) masked damage (**b**) masked dirt (**c**) masked unknown (**d**) masked gap

(**e**) masked dent (**f**) masked crush (**g**) masked scratch (**h**) masked slant

Figure 4. Masked support images of eight classes.

4.1. Experimental Settings

As illustrated in Figure 5, we follow the widely-applied training paradigm in [1,40,41]. The backbone network as well as the detector heads is pretrained on MS COCO. During fine-tuning, all defect images excluding the manually selected support ones are divided into three parts: a training set, a validation set and a test set. Specifically, the training set contains 280 images, the validation set contains 80 images, and the test set contains 40 images. The training set is used to train the model, while the validation set is used to evaluate its detection performance. After training, the test performance of the model that performs best on validation set is evaluated on the test set.

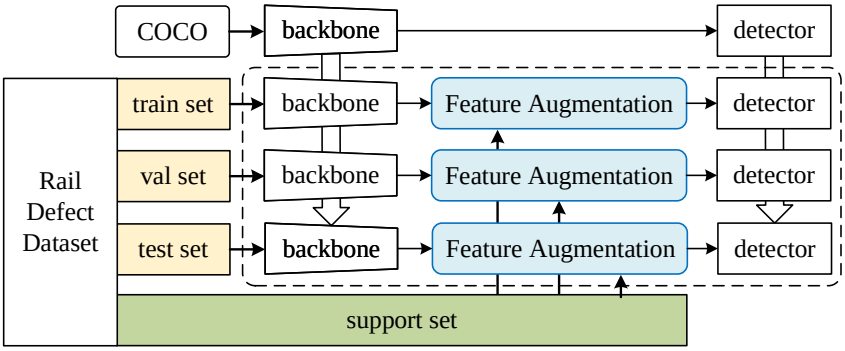


Figure 5. Demonstration of learning strategy of our detection framework. The backbone is pretrained on MS COCO, and fine-tuned on the rail defect dataset together with our proposed feature augmentation module.

All learnable parameters, including the parameters of the backbone feature extractor and the feature augmentation framework are jointly tuned by stochastic gradient descent (SGD) for 500 epochs. The momentum and weight decaying factor are set to be 0.9 and 5×10^{-4}, respectively. All the images are resized to 640×640 pixels before training and testing. It takes about 61.2 h to train the proposed model with 400 images (batch size is 9 with 1 sample image and 8 support images) on one NVIDIA RTX 3090 GPU.

4.2. Detection Results

We first analyze the detection results on our collected rail defect dataset. We compare the accuracy of our approach to YOLO [5], Faster R-CNN [4], AL-MDN [37], and DyHead [38] on both 8-class and 4-class detection tasks.

In Tables 2 and 3, we summarize the results of the experiments performed on eight classes. In the table, numbers in bold are models with the minimum performance indices, 'R' in the backbone column is short for ResNet, 's/s6' refer to the small version of YOLOv5 without/with 4 output layers, and 'F R50/R101' refer to Faster R-CNN with ResNet 50 or ResNet 101. As shown, our proposed method performs better according to mAP@.5 and mAP@.5.95 on both the validation set and test set. The best performance is highlighted as a bold text. The mAP@.5 of our model (i.e., YOLOv5s6 with FA) on validation set is 11.32% better than YOLOv5s6 and 4.27% better than Faster R-CNN R101. Although feature augmentation calculation heavily reduces the detection speed, our proposed model can still achieve real-time detection with more than 30 fps. As a comparison, AL-MDN and DyHead also outperform traditional one-stage and two-stage baselines, and are competitive with our proposed method.

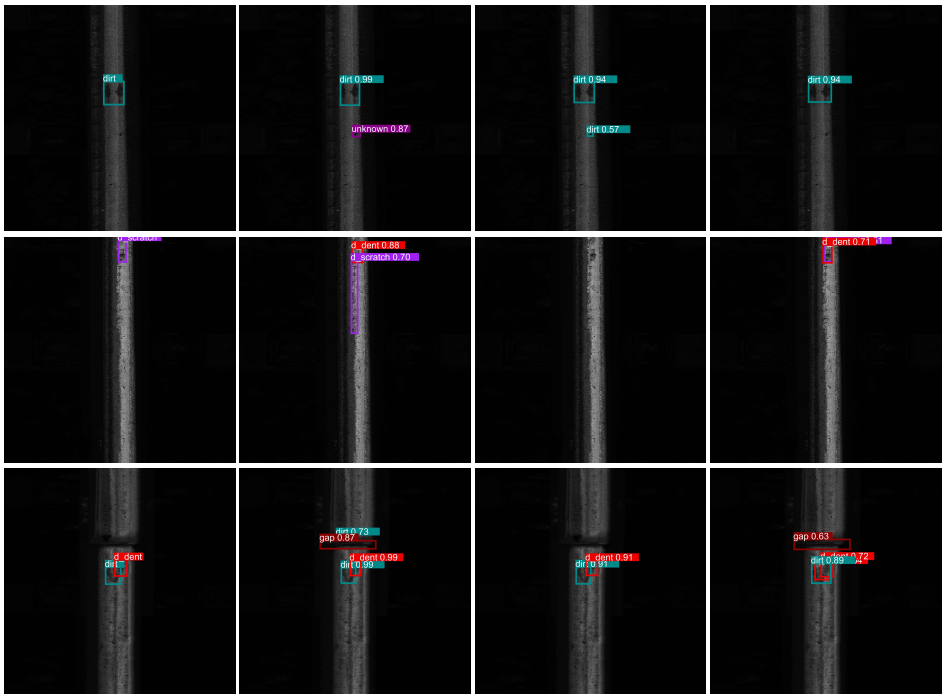
Table 2. Detection results of different methods over the rail defect validation dataset.

Model	Backbone	mAP@.5	mAP@.5:.95
8 defect classes			
YOLOv5 + FA (ours)	s	0.8462	0.3724
YOLOv5 + FA (ours)	s6	0.8643	**0.4232**
YOLOv6 + FA (ours)	s	**0.8714**	0.3911
YOLOv5 [42]	s	0.8371	0.3206
YOLOv5 [42]	s6	0.8316	0.3476
YOLOv6 [43]	s	0.8417	0.3595
Faster R-CNN [4]	R50	0.8351	0.3447
Faster R-CNN [4]	R101	0.8402	0.3366
AL-MDN [37]	VGG16	0.8545	0.3772
DyHead [38]	RetinaNet	0.8513	0.3817
DyHead [38]	F R50	0.8527	0.3833
DyHead [38]	F R101	0.8535	0.3851
4 defect classes			
YOLOv5 + FA (ours)	s	0.8856	0.4557
YOLOv5 + FA (ours)	s6	**0.8992**	0.4979
YOLOv6 + FA (ours)	s	0.8878	**0.5011**
YOLOv5 [42]	s	0.8989	0.4622
YOLOv5 [42]	s6	0.8966	0.4659
YOLOv6 [43]	s	0.8981	0.4730
Faster R-CNN [4]	R50	0.8779	0.4642
Faster R-CNN [4]	R101	0.8916	0.4760
AL-MDN [37]	VGG16	0.8947	0.4861
DyHead [38]	RetinaNet R50	0.8941	0.4903
DyHead [38]	F R50	0.8965	0.4967
DyHead [38]	F R101	0.8932	0.4839

Several defect detection results are shown in Figure 6. In the first row, Faster R-CNN detected an extra unknown object (a false alarm) while YOLOv5 treat the extra object to be a dirt. In the second row, the ground truth is a scratch but it can also be regarded as a dent (i.e., an ambiguous defect). Faster R-CNN identified a dent and a scratch with large position deviation. YOLOv5 failed to detect any object but with the help of feature augmentation it can identify the scratch very close to the ground truth, and also detected an extra dent. In the third row, the ground truth contained a labeled dent, a labeled dirt, and an unlabeled gap. Faster R-CNN and our model can identify the unlabeled gap, while Faster R-CNN mistakenly identified an extra dirt. The precision-recall curves are illustrated in Figure 7.

Table 3. Detection results of different methods over the rail defect test dataset.

Model	Backbone	mAP@.5	mAP@.5:.95	FPS
8 defect classes				
YOLOv5 + FA (ours)	s	**0.757**	0.365	34.36
YOLOv5 + FA (ours)	s6	0.726	0.369	34.25
YOLOv6 + FA (ours)	s	0.740	**0.377**	52.99
YOLOv5 [42]	s	0.657	0.317	52.36
YOLOv5 [42]	s6	0.680	0.346	48.54
YOLOv6 [43]	s	0.674	0.325	72.51
Faster R-CNN [4]	R50	0.719	0.333	8.49
Faster R-CNN [4]	R101	0.726	0.322	9.00
AL-MDN [37]	VGG16	0.744	0.362	5.98
DyHead [38]	RetinaNet R50	0.741	0.367	38.41
DyHead [38]	F R50	0.741	0.367	8.23
DyHead [38]	F R101	0.741	0.367	7.36
4 defect classes				
YOLOv5s + FA (ours)	s	0.796	0.357	41.49
YOLOv5s6 + FA (ours)	s6	0.818	0.391	32.79
YOLOv6 + FA (ours)	s	**0.835**	**0.393**	54.31
YOLOv5 [42]	s	0.816	0.336	56.34
YOLOv5 [42]	s6	0.804	0.386	56.18
YOLOv6 [43]	s	0.827	0.383	72.33
Faster R-CNN [4]	R50	0.808	0.387	14.48
Faster R-CNN [4]	R101	0.820	0.373	9.76
AL-MDN [37]	VGG16	0.816	0.381	6.27
DyHead [38]	RetinaNet R50	0.802	0.377	40.67
DyHead [38]	F R50	0.809	0.381	7.75
DyHead [38]	F R101	0.810	0.389	6.51



(**a**) ground truth (**b**) Faster R-CNN (**c**) YOLO (**d**) ours

Figure 6. Detection result concerning 8 classes. The four columns corresponds to (**a**) ground truth, (**b**) Faster R-CNN-R101, (**c**) YOLOv5s6, and (**d**) YOLOv5s6 with the proposed FA. Three rows are three different test images.

In industrial applications, dirt and gap objects can be ignored, whereas unknown objects are usually regarded as potential defects and should be double-checked manually. Therefore, false detection may cause additional manual inspection time. The detection result revealed that Faster R-CNN is too sensitive and confident to identify many objects that did not exist or need to be ignored. On the contrary, YOLO missed some objects and is too rigid to doubt the training samples. The feature augmentation framework can give scores to help balance the sensitivity and robustness of the model. It can increase the confidence of objects with obvious features but not correctly labeled (e.g., the unlabeled gap in the third row of Figure 6), while reduce the confident of objects with features similar to certain class but are 'far' to selected support defects (e.g., small objects in the first row of Figure 6 that should be ignored).

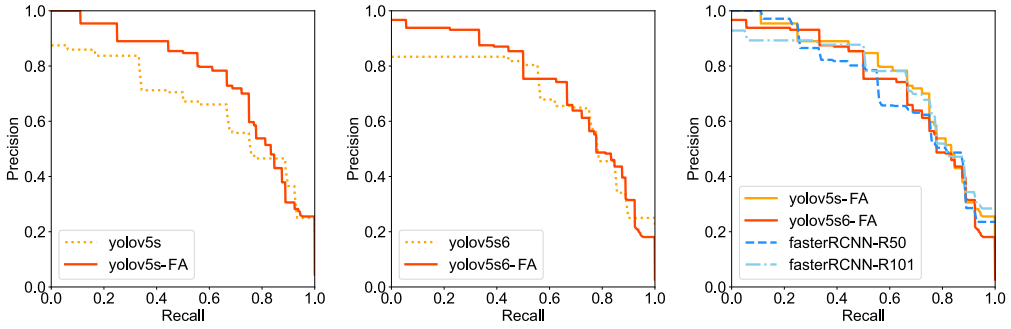
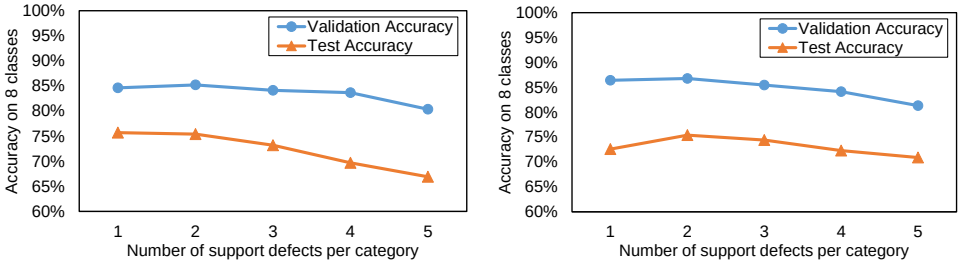


Figure 7. Three group of PR curves, from top to bottom are PR curves of: YOLOv5s and YOLOv5s with FA, YOLOv5s6 and YOLOv5s6 with FA, and two YOLO models with FA compared with two Faster R-CNN models.

4.3. Ablation Study

We conduct a series of ablation studies to analyze the effectiveness of our proposed method. All ablation studies are conducted on the 8-class rail defect test dataset if not otherwise stated. All results are averaged over 10 random runs.

Impact of multiple support defects. We test our method on a different number of support defects per category. We randomly select 2, 3, 4, and 5 support defects in each category, and train our model. The validation and test results of our model are shown in Figure 8. As can be seen from the figure, although the validation performance of the model is slightly improved when using two support defects per category (e.g., 0.7% improvement of YOLOv5s, from 84.62% to 85.21%), the overall performance trend of the model decreases with the increase of the number of support defects. This result indicates that although more support defects can introduce more diverged defect patterns to the model, it cannot help improve the detection performance of the model. Instead, the redundant defect patterns may confuse the model and impede the detection process.



(**a**) YOLOv5s (**b**) YOLOv5s6

Figure 8. Effect of different number of support defects per category on validation and test performance of (**a**) YOLOv5s with FA and (**b**) YOLOv5s6 with FA.

Impact of support feature generation methods. Support class prototypes are usually generated by averaging all support images of each class [20,44,45]. In this section, we analyze the effectiveness of two support class prototype feature generation methods other than averaging: random selection and summation and normalization (add and norm). Random selection means to randomly select one of the support feature for subsequent feature augmentation, while summation and normalization apply normalization after adding all the support features. We further set the number of support defects per category to be 2. The detection results are shown in Table 4. As shown in the table, the model perform best by averaging all support images. However, the model does not degrade much when adopting random selection or by averaging only two support images, but the amount of calculation during training is greatly reduced.

Table 4. Evaluation of support feature generation approaches.

Method	Feature Generation Approach	mAP@.5	mAP@.5:.95
YOLOv5s + FA	random selection	0.758	0.371
YOLOv5s + FA	random selection (2)	0.757	0.369
YOLOv5s + FA	add & norm	0.743	0.356
YOLOv5s + FA	add & norm (2)	0.729	0.344
YOLOv5s + FA	average	**0.760**	**0.372**
YOLOv5s + FA	average (2)	0.757	0.366

Impact of Mask. We studied the effect of the mask operation within the detection process. The detection results are shown in Table 5. From the two tables we can see that the detection performance of the model decreases slightly without mask operation. This result is easy to understand, since the mask operation can effectively remove the background information interference of non-defect parts. It can help the model to focus more on useful objects rather than the background.

Table 5. Impact of mask.

Method	w/ Mask		w/o Mask	
	mAP@.5	mAP@.5:.95	mAP@.5	mAP@.5:.95
YOLOv5s + FA	**0.757**	0.365	0.748	0.345
YOLOv5s6 + FA	0.726	**0.369**	0.716	0.354

Generalization on existing backbones. We evaluate the generalization ability of our proposed feature augmentation method by plugging it to popular object detection backbones other than YOLO, such as Faster R-CNN and RetinaNet [27]. These two backbones are typical two-stage and one-stage object detection frameworks. As shown in Table 6, our proposed feature augmentation module boosts the two backbones by around 1% mAP@.5. It demonstrates the generality of our method. Yet we still prefer our module to be plugged to one-stage object detection frameworks for faster detection ability.

Table 6. Generalization of FA module applied to object detection backbones.

Method	Backbone	mAP@.5	mAP@.5:.95	FPS
Faster R-CNN [4]	R50	0.719	0.333	8.49
Faster R-CNN [4] + FA	R50	0.727	0.346	8.13
RetinaNet [46]	R50	0.687	0.328	24.26
RetinaNet [46] + FA	R50	0.696	0.337	20.98
SSD [27]	VGG16	0.684	0.321	21.78
SSD [27] + FA	VGG16	0.698	0.342	19.91

5. Discussion

In this paper, we focused on the rail defect detection task and developed a feature augmentation framework to ensure fast and accurate defect detection. The proposed method is especially designed to take advantage of the limited but precisely annotated support defect images. Multi-scale defect features were aggregated to calculate a learnable similarity metric between sample image and support defect images. Support features are then weighed by the similarity and concatenated with the input sample feature to obtain the final feature map. The feature augmentation framework can help increase the detecting accuracy concerning multiple distinctive defect types, and reduce the confidence of small defects that should be ignored. Experimental results validated the proposed method and showed its potential usefulness in practice.

To test the proposed method, we annotated and constructed a novel rail defect dataset. All the rail defection images are captured from the 9 km railway test loop built by the National Academy of Railway Sciences Test Center. As shown experimentally, the proposed framework outperforms the two baselines in terms of detection accuracy on both validation and test rail defect dataset concerning eight classes. Our method is capable of the inspection task at the running speed of 160 km per hour.

Our pixel-wise attention within the feature augmentation framework can be treated as a combination of self attention and cross attention, since we retrieve hidden representations from both the sample feature and the support features.The feature augmentation framework can give scores to help balance the sensitivity and robustness of the model. It can increase the confidence of objects with obvious features but not correctly labeled, while reduce the confident of objects with features similar to certain class but are 'far' to selected support defects.

We consider as possible future works to investigate the use of our framework in other defect detection tasks where there are many kinds of defect types but relatively fewer defect samples. The combination of line scan cameras and laser cameras could also be an applicable research direction. For example, an extra 3D laser camera could provide height value for better detection when the budget and hardware performance allow. The height values provided by laser cameras can help better eliminate disturbances such as dirt [47–49], dust and plants, while the x-y images provided by line scan cameras can help detect various damage types intuitively.

Author Contributions: Conceptualization, P.D.; methodology, H.L. (Hongjue Li); software, H.L. (Hongjue Li); validation, H.L. (Hongjue Li), H.L. (Hailang Li) and Z.H.; formal analysis, Z.H.; investigation, H.S.; resources, H.S.; data curation, H.S.; writing—original draft preparation, H.L. (Hongjue Li); writing—review and editing, H.L. (Hongjue Li) and H.L. (Hongjue Li); visualization, J.L.; supervision, P.D.; project administration, P.D. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by the High-speed Rail Joint Fund Projects grant number U1934215, the Scientific Research Projects of China Academy of Railway Sciences grant number 2020YJ065, and the Science and Technology R&D Program of China Railway Corporation grant number K2021T015.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the corresponding author.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 580–587.
2. Lin, T.Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature pyramid networks for object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 2117–2125.
3. Liu, S.; Qi, L.; Qin, H.; Shi, J.; Jia, J. Path aggregation network for instance segmentation. In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 8759–8768.
4. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. *Adv. Neural Inf. Process. Syst.* **2015**, *28*, 91–99. [CrossRef] [PubMed]
5. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
6. Lin, T.Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft coco: Common objects in context. In Proceedings of the European Conference on Computer Vision, Zurich, Switzerland, 6–12 September 2014; Springer: Berlin/Heidelberg, Germany, 2014; pp. 740–755.
7. Deng, J.; Dong, W.; Socher, R.; Li, L.J.; Li, K.; Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; pp. 248–255.
8. Wei, X.; Yang, Z.; Liu, Y.; Wei, D.; Jia, L.; Li, Y. Railway track fastener defect detection based on image processing and deep learning techniques: A comparative study. *Eng. Appl. Artif. Intell.* **2019**, *80*, 66–81. [CrossRef]
9. Hinterstoisser, S.; Lepetit, V.; Wohlhart, P.; Konolige, K. On pre-trained image features and synthetic images for deep learning. In Proceedings of the European Conference on Computer Vision (ECCV) Workshops, Munich, Germany, 8–14 September 2018.
10. Varma, M.; Lu, M.; Gardner, R.; Dunnmon, J.; Khandwala, N.; Rajpurkar, P.; Long, J.; Beaulieu, C.; Shpanskaya, K.; Fei-Fei, L.; et al. Automated abnormality detection in lower extremity radiographs using deep learning. *Nat. Mach. Intell.* **2019**, *1*, 578–583. [CrossRef]
11. Tajbakhsh, N.; Shin, J.Y.; Gurudu, S.R.; Hurst, R.T.; Kendall, C.B.; Gotway, M.B.; Liang, J. Convolutional neural networks for medical image analysis: Full training or fine tuning? *IEEE Trans. Med. Imaging* **2016**, *35*, 1299–1312. [CrossRef] [PubMed]
12. Mnih, V.; Heess, N.; Graves, A. Recurrent models of visual attention. In *Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Red Hook, NY, USA, 2014; pp. 2204–2212.
13. Bahdanau, D.; Cho, K.; Bengio, Y. Neural machine translation by jointly learning to align and translate. *arXiv* **2014**, arXiv:1409.0473.
14. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems*; NeurIPS: San Diego, CA, USA, 2017; pp. 5998–6008.
15. Hou, R.; Chang, H.; Ma, B.; Shan, S.; Chen, X. Cross attention network for few-shot classification. *arXiv* **2019**, arXiv:1910.07677.
16. Hu, H.; Gu, J.; Zhang, Z.; Dai, J.; Wei, Y. Relation Networks for Object Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018.
17. Kang, B.; Liu, Z.; Wang, X.; Yu, F.; Feng, J.; Darrell, T. Few-shot object detection via feature reweighting. In Proceedings of the IEEE International Conference on Computer Vision, Seoul, Korea, 27 October–2 November 2019; pp. 8420–8429.
18. Fan, Q.; Zhuo, W.; Tang, C.K.; Tai, Y.W. Few-shot object detection with attention-RPN and multi-relation detector. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 4013–4022.
19. Yan, X.; Chen, Z.; Xu, A.; Wang, X.; Liang, X.; Lin, L. Meta r-cnn: Towards general solver for instance-level low-shot learning. In Proceedings of the IEEE International Conference on Computer Vision, Seoul, Korea, 27 October–2 November 2019; pp. 9577–9586.
20. Hu, H.; Bai, S.; Li, A.; Cui, J.; Wang, L. Dense Relation Distillation with Context-aware Aggregation for Few-Shot Object Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Virtual, 19–25 June 2021; pp. 10185–10194.
21. Lienhart, R.; Maydt, J. An extended set of haar-like features for rapid object detection. In Proceedings of the International Conference on Image Processing, Rochester, NY, USA, 22–25 September 2002; Volume 1, p. 1.
22. Mizuno, K.; Terachi, Y.; Takagi, K.; Izumi, S.; Kawaguchi, H.; Yoshimoto, M. Architectural study of HOG feature extraction processor for real-time object detection. In Proceedings of the 2012 IEEE Workshop on Signal Processing Systems, Quebec City, QC, Canada, 17–19 October 2012; pp. 197–202.
23. Hastie, T.; Rosset, S.; Zhu, J.; Zou, H. Multi-class adaboost. *Stat. Its Interface* **2009**, *2*, 349–360. [CrossRef]
24. Guo, M.; Zhao, Y.; Xiang, J.; Zhang, C.; Chen, Z. Review of object detection methods based on SVM. *Control Decis.* **2014**, *29*, 193–200.
25. Uijlings, J.R.; Van De Sande, K.E.; Gevers, T.; Smeulders, A.W. Selective search for object recognition. *Int. J. Comput. Vis.* **2013**, *104*, 154–171. [CrossRef]
26. Zitnick, C.L.; Dollár, P. Edge boxes: Locating object proposals from edges. In Proceedings of the European Conference on Computer Vision, Zurich, Switzerland, 6–12 September 2014; Springer: Berlin/Heidelberg, Germany, 2014; pp. 391–405.
27. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. Ssd: Single shot multibox detector. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016; Springer: Berlin/Heidelberg, Germany, 2016; pp. 21–37.

28. Durand, T.; Mordan, T.; Thome, N.; Cord, M. WILDCAT: Weakly Supervised Learning of Deep ConvNets for Image Classification, Pointwise Localization and Segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 642–651.
29. Pan, X.; Ren, Y.; Sheng, K.; Dong, W.; Yuan, H.; Guo, X.; Ma, C.; Xu, C. Dynamic refinement network for oriented and densely packed object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 11207–11216.
30. Xu, B.; Liang, H.; Liang, R.; Chen, P. Locate Globally, Segment Locally: A Progressive Architecture With Knowledge Review Network for Salient Object Detection. In Proceedings of the AAAI Conference on Artificial Intelligence, Virtual, 2–9 February 2021; Volume 35, pp. 3004–3012.
31. Sun, L.; Zhao, C.; Yan, Z.; Liu, P.; Duckett, T.; Stolkin, R. A novel weakly-supervised approach for RGB-D-based nuclear waste object detection. *IEEE Sens. J.* **2018**, *19*, 3487–3500. [CrossRef]
32. Ouyang, W.; Wang, X.; Zhang, C.; Yang, X. Factors in finetuning deep model for object detection with long-tail distribution. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 864–873.
33. Shorten, C.; Khoshgoftaar, T.M. A survey on image data augmentation for deep learning. *J. Big Data* **2019**, *6*, 1–48. [CrossRef]
34. Chen, Y.C.; Zhu, X.; Zheng, W.S.; Lai, J.H. Person re-identification by camera correlation aware feature augmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *40*, 392–408. [CrossRef] [PubMed]
35. Chen, Z.; Fu, Y.; Zhang, Y.; Jiang, Y.G.; Xue, X.; Sigal, L. Multi-level semantic feature augmentation for one-shot learning. *IEEE Trans. Image Process.* **2019**, *28*, 4594–4605. [CrossRef] [PubMed]
36. Northcutt, C.G.; Athalye, A.; Mueller, J. Pervasive label errors in test sets destabilize machine learning benchmarks. *arXiv* **2021**, arXiv:2103.14749.
37. Choi, J.; Elezi, I.; Lee, H.J.; Farabet, C.; Alvarez, J.M. Active Learning for Deep Object Detection via Probabilistic Modeling. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Virtual, 11–17 October 2021; pp. 10264–10273.
38. Dai, X.; Chen, Y.; Xiao, B.; Chen, D.; Liu, M.; Yuan, L.; Zhang, L. Dynamic Head: Unifying Object Detection Heads with Attentions. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 7373–7382.
39. Wu, Y.; Kirillov, A.; Massa, F.; Lo, W.Y.; Girshick, R. Detectron2. 2019. Available online: https://github.com/facebookresearch/detectron2 (accessed on 20 June 2021).
40. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask r-cnn. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2961–2969.
41. Chen, Y.; Yang, T.; Zhang, X.; Meng, G.; Xiao, X.; Sun, J. Detnas: Backbone search for object detection. *Adv. Neural Inf. Process. Syst.* **2019**, *32*, 6642–6652.
42. ultralytics. YOLOv5. 2020. Available online: https://github.com/ultralytics/yolov5 (accessed on 7 September 2021).
43. ultralytics. YOLOv6. 2022. Available online: https://github.com/meituan/YOLOv6 (accessed on 30 July 2022).
44. Li, B.; Yang, B.; Liu, C.; Liu, F.; Ji, R.; Ye, Q. Beyond Max-Margin: Class Margin Equilibrium for Few-shot Object Detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 7363–7372.
45. Han, G.; He, Y.; Huang, S.; Ma, J.; Chang, S.F. Query adaptive few-shot object detection with heterogeneous graph convolutional networks. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Nashville, TN, USA, 20–25 June 2021; pp. 3263–3272.
46. Lin, T.Y.; Goyal, P.; Girshick, R.; He, K.; Dollár, P. Focal loss for dense object detection. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2980–2988.
47. Santur, Y.; Karaköse, M.; Akin, E. A new rail inspection method based on deep learning using laser cameras. In Proceedings of the 2017 International Artificial Intelligence and Data Processing Symposium (IDAP), Malatya, Turkey, 16–17 September 2017; pp. 1–6.
48. Santur, Y.; Karaköse, M.; Akın, E. Condition monitoring approach using 3D modelling of railway tracks with laser cameras. In Proceedings of the International Conference on Advanced Technology & Sciences (ICAT'16), Konya, Turkey, 1–3 September 2016; Volume 132, p. 135.
49. Santur, Y.; Karaköse, M.; Akın, E. Learning based experimental approach for condition monitoring using laser cameras in railway tracks. *Int. J. Appl. Math. Electron. Comput.* **2016**, 1–5. [CrossRef]