Hindawi
Structural Control and Health Monitoring
Volume 2024, Article ID 1898088, 16 pages
https://doi.org/10.1155/2024/1898088

Research Article

DSNet: A Computer Vision-Based Detection and Corrosion Segmentation Network for Corroded Bolt Detection in Tunnel

Lei Tan[1,2] **Xiaohan Chen**[3] **Dajun Yuan**[4] **and Tao Tang**[1,3]

[1]*State Key Laboratory of Advanced Rail Autonomous Operation, Beijing Jiaotong University, Beijing 100044, China*
[2]*Beijing Municipal Engineering Research Institute, Beijing 100037, China*
[3]*School of Electronic and Information Engineering, Beijing Jiaotong University, Beijing 100044, China*
[4]*School of Civil Engineering, Beijing Jiaotong University, Beijing 100044, China*

Correspondence should be addressed to Lei Tan; tanlei@bjtu.edu.cn

Received 20 July 2023; Revised 3 September 2023; Accepted 19 September 2023; Published 13 February 2024

Academic Editor: Ka-Veng Yuen

Corroded bolt detection has been confirmed as a major issue in the structure health monitoring (SHM) of tunnels. However, detection-only methods will miss the corroded bolts, arising from the small rust area. In this study, the task is divided ingeniously into two parallel tasks, i.e., bolt detection and pixel-level rust segmentation, and the objective is fulfilled by taking the intersection of the two tasks, with the aim of enhancing the performance. To be specific, a detection and segmentation network (DSNet) is proposed based on multitask learning, leading to reduced false and missed detection rates. The coordinate attention module enhancing the focus of bolts in tunnel patches is incorporated in the detection branch, and the cross-stage partial-based decoder which can more accurately determine whether a pixel pertains to the corrosion area is employed in the segmentation branch. The mentioned branches share the same backbone to simplify the model. Sufficient comparisons and ablation experiments are performed to prove the superiority of the proposed algorithm based on the corroded bolt dataset captured from a real subway tunnel, which is publicly available in https://github.com/StreamHXX/Tunnel-lining-disease-image.

1. Introduction

Bolts adopted to fasten the linings built on the surface of a subway tunnel to prevent the metro trains from rock and soil falls [1]. Since bolts are exposed to the open air, the above-mentioned bolts will be subjected to corrosion, thus threatening the tightness of linings. However, inspection and maintenance can be only conducted in a nonrunning period of about three hours, during which a trained maintenance team of 10 people can only check nearly three kilometers [2, 3]. Under manual inspection with low efficiency, high missed detection, and high false alarms, researchers are forced to develop an automatic inspection method with prominent performance.

Computer vision (CV) has aroused wide attention for its simplicity in deployment and outstanding cost-effectiveness. Traditional CV algorithms rely on hand-crafted feature extraction and hard-coded algorithms [4]. Besides, the robustness of the above-mentioned algorithms is contingent on the features they have been programmed to identify. Moreover, they may exhibit low performance under lighting, orientation, and viewpoint variations, limiting the applicability of automatic defect inspection in the tunnel [5]. For the deep learning-based CV, convolutional neural networks (CNNs) dominate in the field of the CV [6] with great success in object detection [7]. Deep learning-based CV has achieved prominent results in device fault detection and measurement (e.g., bolts) over the past few years [8–12].

The difficulty of corroded bolt detection based on CV is manifested as high false alarms and high missed detection because the characteristics of the corroded area may vanish or be ignored after the captured images are fed into the CNNs, especially the convolution and pooling operations due to the small rust regions.

On that basis, a pixel-level corrosion detection method [13, 14] should be developed to address the problem. Moreover, semantic segmentation associating a label or category with every pixel in an image is capable of capturing the details in the image [15]. Accordingly, the corroded bolt detection task is divided creatively into two parallel tasks, i.e., bolt detection and pixel-level rust segmentation. Next, the intersection of the two tasks is taken. To be specific, a detection and segmentation network (DSNet) based on multitask learning is developed to lessen missed detection and fault alarms. The detection subnetwork employs the coordinate attention module (CAM) that enhances the focus of bolts in tunnel images, while the segmentation subnetwork applies a cross-stage partial (CSP)-based decoder to increase the accuracy of detecting whether a pixel falls under the corrosion area. Although the segmentation branch is effective in distinguishing the rust at the pixel level, it will mistakenly label corrosion in nonbolt regions, which is not desirable. Thus, the corrosion segmentation results should be limited to the range of the bolt in an image.

Consequently, we make the above two subnetworks work in parallel and thus propose a dual multitask learning algorithm, the DSNet, which realizes bolt detection and pixel-level segmentation in tunnel lining images. It is noted that although it is feasible to connect the two parts in series, this approach necessitates the use of two backbones and entails significantly greater computational requirements. Given the need for less inference time, the two subnetworks share an identical backbone to simplify the model. The main contributions of this study are elucidated as follows.

(i) We propose a lightweight and accurate detection and segmentation algorithm for tunnel corroded bolts. The proposed multitask model DSNet can simultaneously accomplish bolt detection and corrosion region segmentation. A multitask result fusion method is also designed to reduce the miss detection rate and false detection rate of corroded bolt detection.

(ii) We incorporate the CAM to improve the YOLOx in the detection branch and design a CSP-based decoder for the segmentation branch to obtain better performance.

(iii) Sufficient ablation and comparative experiments are performed on the data collected from a practical tunnel in Beijing to verify the advantages of our method. Our dataset is publicly available at https://github.com/StreamHXX/Tunnel-lining-disease-image.

The rest of this paper is organized as follows. Section 2 reviews the related work. Section 3 describes our proposed algorithm. Section 4 shows detailed information on experimental settings, evaluation metrics, results, and analysis and explanations. Finally, Section 5 concludes our main work.

2. Related Work

Before the related work, it is necessary to give acronyms, and typical models mentioned in the paper are shown in Table 1.

2.1. Bolt Detection. Compared with the traditional CV-based methods that rely on manual filter designs [25–27], the deep learning-based CV of bolt detection has been extensively employed in the engineering field for its superior performance in terms of accuracy, robustness, and generalization ability. The existing mainstream deep learning-based CV of bolt detection can fall into two-stage models and one-stage models (i.e., end-to-end models). The "one-stage" and the "two-stage" methods mainly describe the workflow of object detection algorithms, more specifically, how these algorithms perform the tasks of object localization (identifying the location of the object in the image) and classification (determining the category of the object). The "two-stage" models refer to two main consecutive steps in these algorithms during the object detection process, while one-stage methods perform object localization and classification in one step. Consequently, two-stage structures require significantly more time for separate training and detection, which hampers the speed of corroded bolt detection work [2].

Regarding the two-stage models, Cha et al. [22] proposed a real-time detection of engineering structure damage used to detect covering concrete cracks, steel and bolt corrosion, and steel delamination based on region-based CNNs (R-CNNs). Huynh et al. [28] put forward a loose bolt detection method based on an R-CNN and used the Hough line transform to estimate the loosening angle. The improved models, mask R-CNN and faster R-CNN of R-CNNs, have been extensively adopted for anomalous (e.g., corrosion, loosening, and loss) bolt detection in engineering structures [5, 29–31]. Nevertheless, these models are subjected to drawbacks such as high computational demands, limited generalization ability, and reduced robustness. Furthermore, two-stage models may be more difficult to deploy for their significantly higher computational requirements, making them slower and less practical for real-world scenarios.

For the one-stage models, the YOLO refers to an end-to-end detection model that benefits from its anchor-free mechanism and simplified end-to-end detection principle [32]. The training and detection speed of the YOLO have been notably increased compared with those of the two-stage model, such that its variant models are widely used in the engineering field [33, 34]. By combining more powerful feature extraction networks, introducing attention mechanisms, and implementing multiscale processing techniques, the YOLO series has achieved better performance while balancing speed and accuracy. Yang et al. [35] introduced a bolt-loosening detection method by combining the manual torque method with various versions of the YOLO, and the experiments showed that the method achieved good experimental indexes with strong application value in the scenario of using smartphones. Tan et al. [2] used the YOLOv5 model to detect and label the corroded bolts in the tunnel whose detection speed and accuracy are far higher than the two-stage model. Although the YOLOv5 has achieved excellent results in various object detection tasks, it still has some limitations. One of the main drawbacks is that it requires significant computational resources, which may limit its applicability to some real-world scenarios.

Table 1: Acronyms and full name correspondence table.

Acronyms	Full name	Characteristic
AP	Average precision	—
BCE	Binary cross entropy	—
CAM	Coordinate attention module [16]	Enhanced sensitivity of feature maps to target direction and location
CBAM	Convolutional block attention module [17]	Effective attention module suitable for all feedforward CNNs
CSP	Cross-stage partial [18]	Splitting inputs to reduce computational effort
CSPDarkNet	Cross-stage partial DarkNet [18]	A backbone network of YOLO series models
DSNet	Detection and segmentation network	The proposed network model
FCN	Fully convolutional networks [19]	Enhancing semantic segmentation using convolutional layers
FPS	Frame per second	—
FAR	False alarm rate	—
GT	Ground truth	—
HRNet	High-resolution net [20]	Parallel connection of subnets and fuse feature maps of the identical depth
IoU	Intersection over union	—
mIoU	Mean intersection over union	—
mPA	Mean pixel accuracy	—
MDR	Missed detection rate	—
PAFPN	Path aggregation feature pyramid network [21]	Strengthen the structure of the feature pyramid
R-CNNs	Region-based CNNs [22]	The first deep learning-based detection algorithm
SE	Squeeze-and-excitation	Models the correlation between feature channels and enhances important features
SPP	Spatial pyramid pooling [23]	Convert input images of different sizes to the same size
U-Net	U-shaped net [24]	Symmetrical U-shaped structure allows for more complete feature integration
TP	True positive	—
TN	True negative	—
FP	False positive	—
FN	False negative	—

Moreover, the YOLOv5 may face performance degradation in detecting small objects, such as bolts in tunnels, due to the constraints of its anchor-based approach. Besides, YOLOv5 has a large down-sampling multiple, making it difficult for deep feature maps to learn the feature information of small targets [36, 37].

2.2. Corroded Area Segmentation. Traditional CV image segmentation methods are usually based on a color and/or texture analysis, requiring manual recognition of common domain characteristics [38]. A relatively simple algorithm can obtain better corrosion detection results since the corrosion feature is red/brown [13, 14, 39]. This type of method is affected by environmental factors, resulting in its unstable accuracy and poor robustness, and then gradually replaced by deep learning-based CV. Furthermore, deep learning-based segmentation models have been widely used in the field of tunnel lining crack identification [40, 41].

Atha and Jahanshahi [42] employed a VGGNet-16 model based on CNNs to process images in a 128×128 sliding window for assessing the corrosion of metal surfaces. While VGGNet-16 can be adapted for semantic segmentation tasks by replacing the fully connected layers with convolutional layers, it suffers from high computational requirements due to a large number of parameters. Dung and Anh [43] proposed a concrete crack detection method based on fully convolutional networks (FCNs) [19] system structure and achieved better results. However, the FCN does not have skip connections that are conducive to capturing more detailed information from the input image; its up-sampling process can trigger a loss of spatial information.

Chen et al. [44] introduced U-Net [45] to corrosion segmentation on steel bridges. The U-Net uses an encoder-decoder architecture with skip connections between the encoder and decoder, which allows the model to capture more detailed information from the input image and generate more accurate segmentation maps. Nevertheless, the U-Net may require more training data and longer training time compared with the FCN. Fondevik et al. [15] built a corrosion dataset for the evaluation of the pyramid scene parsing network and a mask R-CNN for semantic segmentation and instant segmentation, respectively. These approaches demonstrated the significant potential of deep learning models in the task of corrosion segmentation. However, as discussed in Section 2.1, R-CNNs have a two-stage structure that results in high computational demands and other limitations for deployment. However, efficient approaches are required for tunnel inspection due to the limited nonrunning period. Thus, a trade-off between speed and accuracy must be considered.

2.3. Current Gaps and Limitations. In summary of Section 2.1 and Section 2.2, the current models face several challenges such as time-consuming, high computational requirements, and difficulties in deployment for tunnel inspection. Also, the current model to reduce MDR and FAR mainly relies on the model itself, which in turn leads to MDR and FAR cannot be effectively controlled. Moreover, there are few studies that focus on both detecting bolts and segmenting corrosion, thus taking on critical significance in crucial for promoting better structural health monitoring of bolts in tunnel linings. Accordingly, we propose a dual multitask approach for tunnel bolt detection and corrosion segmentation to bridge these gaps.

3. Methodology

The proposed method takes an image captured in a tunnel with a resolution of 640×640 as input. Utilizing the concept of multitask learning, our proposed model employs a parallel architecture to execute both detection and segmentation tasks simultaneously. This design enables the detection and segmentation branches of the model to share features extracted from the backbone, allowing the model to address both tasks in a single pass. Accordingly, the need for separate models to handle each task is eliminated, thereby improving the efficiency of tunnel inspection.

The illustration shown in Figure 1 provides an overview of the system for detecting tunnel bolts and segmenting corrosion areas based on YOLOx. The system consists of two primary components: the tunnel bolt detection module and the corrosion area segmentation module. The input for the system is the scanned image of the tunnel and is processed by the backbone of DSNet. This backbone extracts global features and subsequently distributes them to both the detection branch and the segmentation branch. The detection branch processes these features further to determine the position of the bolts in the tunnel and generates an output that includes the bolt location information. The segmentation branch of the system employs features extracted from the backbone as encoded features. The decoder in the segmentation branch restores the resolution of the feature map through an up-sampling operation, such that a mask is generated, which can then be utilized to identify and highlight the corrosion areas on the bolts. The final outcome of the system refers to a comprehensive result that integrates the bolt detection result and the corrosion area segmentation mask, comprehensively representing the location of bolts and the extent of corrosion.

3.1. Model Architecture

3.1.1. Structure of Backbone. Figure 2 shows the overall architecture of the backbone. CBR represents the operation of convolution + batch normalization + ReLU. The backbone network effectively learns to extract meaningful representations and filter independent information by applying a series of convolutional operations. It applies filters of different sizes and depths to capture spatial patterns, edges, textures, and other relevant visual cues that contribute to detecting objects. While the process of feature extraction through the backbone network may lead to the loss of certain features, it remains indispensable for enabling subsequent network structures to capture the intricate details found in real-world images. This significance becomes particularly pronounced in scenarios where the network architecture is relatively shallow. Through the utilization of the backbone

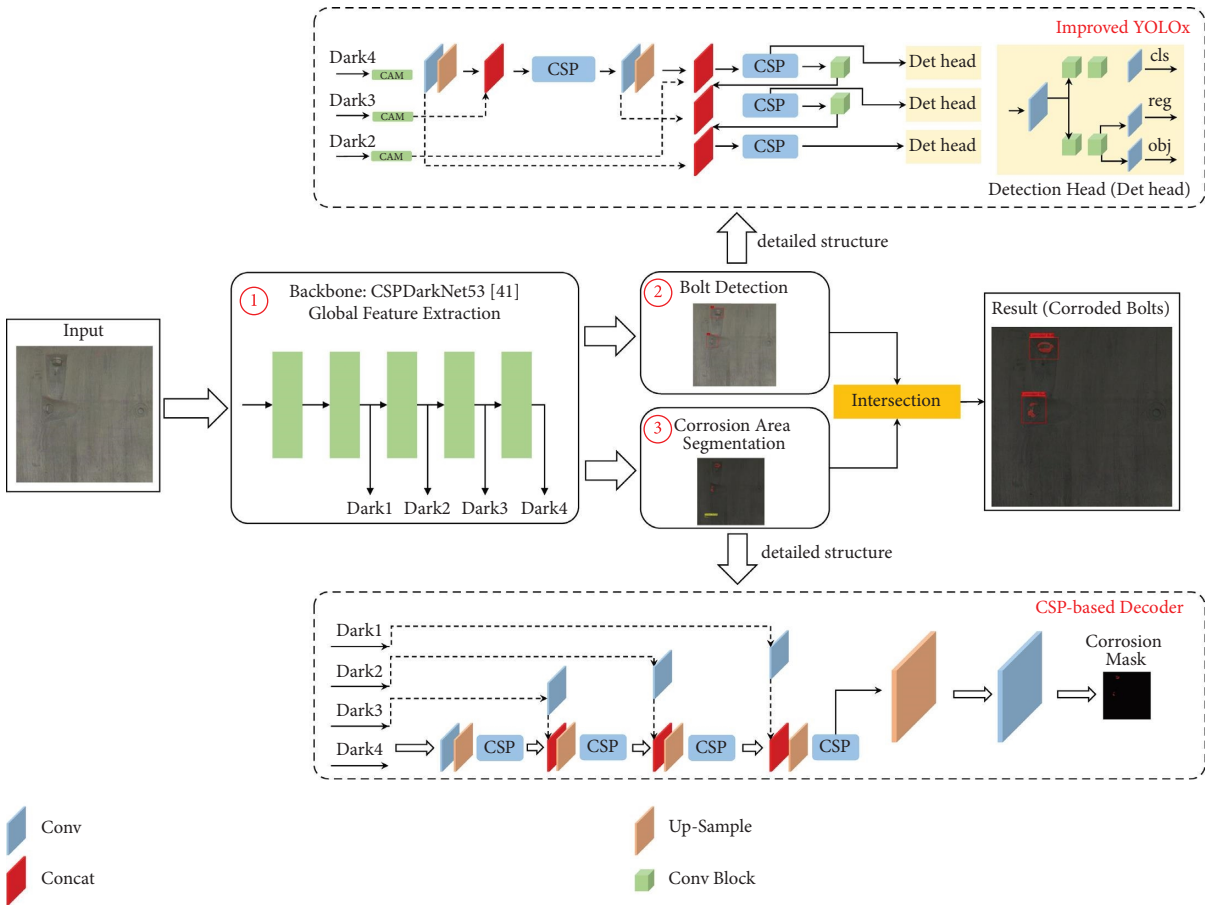

FIGURE 1: An overview of the tunnel bolt positioning and corrosion area segmentation system. The CSPDarkNet53-based backbone ① first extracts features from the image. Then, the bolt detection branch ② and the corrosion area segmentation branch ③ further process the feature independently and complete the task of bolt detection and corrosion area segmentation. Finally, the model ensembles the results from the two branches and gives a final result of corroded bolt detection.

network, object detection algorithms can harness its capacity to extract and encode intricate features. The above-mentioned enhanced features enable more accurate classification and precise localization of objects in an image. The backbone network's capability to capture both low-level and high-level features contributes to the overall performance and robustness of the object detection system.

Following the demands of practical applications, the backbone of the network must exhibit the dual characteristics of lightweights to meet the requirement of fast inference speed and robust feature extraction capability to ensure high accuracy. The CSP architecture has proven to augment the learning capacity of convolutional neural networks, concurrently upholding accuracy, reducing computational demands, minimizing memory utilization, and reducing network weight [18]. Consequently, the CSPDarkNet architecture functions as the foundational framework for the proposed model.

In the CSP approach, the input to the network block is divided into two parts before its processing. One of these parts is processed by the original block, while the other part undergoes a direct shortcut operation. Subsequently, the two parts are combined and result in the final output of the block. Figure 3 illustrates the operation flow of the CSP. The CSP structure is incorporated into every residual block in the DarkNet framework in the CSPDarkNet, allowing for a reduction in the computational complexity of DarkNet while preserving its accuracy performance.

Moreover, the YOLOv4 algorithm [23] incorporates a spatial pyramid pooling (SPP) structure at the end of the CSPDarkNet architecture with the aim of expanding its receptive field. The SPP structure pools the final output of the CSPDarkNet at kernels of 5×5, 9×9, and 13×13, respectively, and concatenates the results with the final output to produce a high-dimensional feature map. Subsequently, a 1×1 convolution operation is applied to reduce the dimensionality of the feature map back to its original size. The operation flow of the SPP is depicted in Figure 4.

With an input image of size 640×640, the backbone of the model will generate four different output sizes, i.e., 160×160, 80×80, 40×40, and 20×20 (represented as Dark1, Dark2, Dark3, and Dark4 in Figures 2 and 1). The detection branch utilizes the outputs from Dark2, Dark3, and Dark4 to determine the location of the bolts in the tunnel. Meanwhile, the segmentation branch decodes the mask and identifies the corrosion areas of the bolts using all the output features.

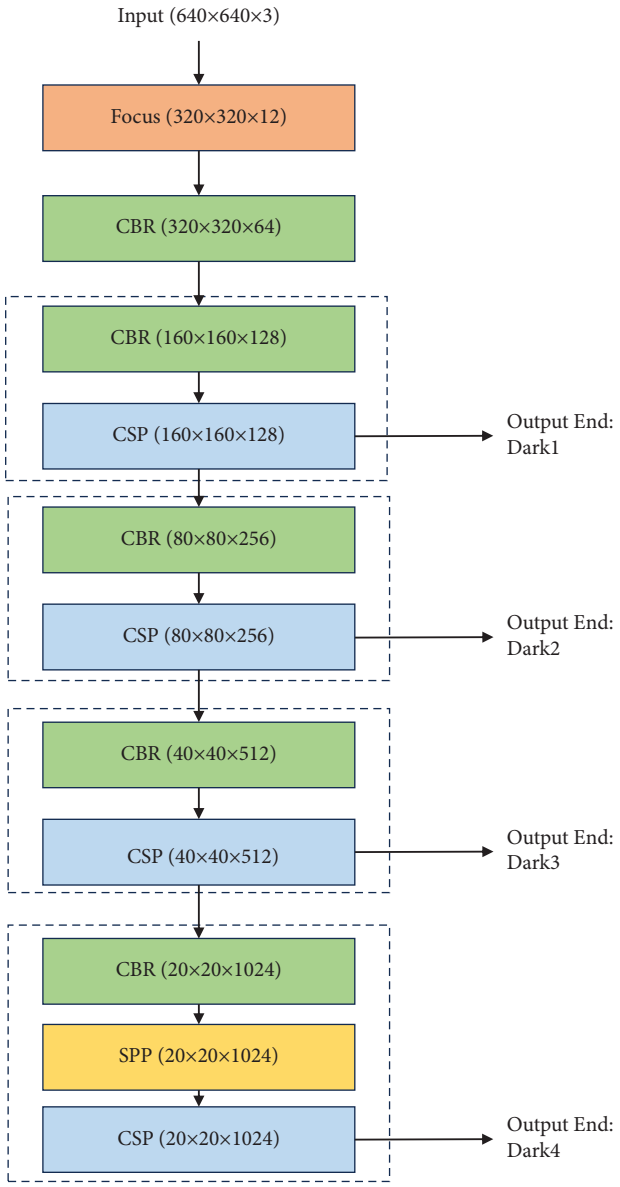

Figure 2: The overall architecture of the backbone. The CSPDarkNet53-based backbone extracts features that can be understood by the following modules from the complex image. It has four output ends (Dark1, Dark2, Dark3, and Dark4) outputting features of different scales.

3.1.2. Bolt Detection. Typically, bolts in images acquired from tunnel scans exhibit limited color contrast compared to their background. This necessitates the enhancement of bolt detection performance by directing the attention of the decoder toward the bolts, thereby augmenting its capability to extract bolt-specific features.

Given the strong local and positional features of bolts, an attention mechanism can be incorporated to enhance the focus of the network on these features. The CAM [16] refers to a novel module that incorporates spatial location information into the channel attention mechanism while maintaining a low computational cost. This makes it suitable for integration into lightweight networks. Figure 5 depicts

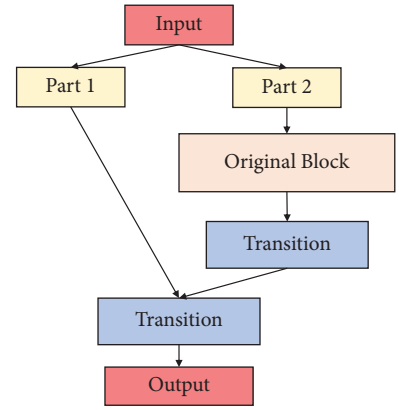

Figure 3: The operation flow of CSP. CSP divides the input into two parts and processes them differently. CSP reduces the computational complexity while preserving the accuracy performance.

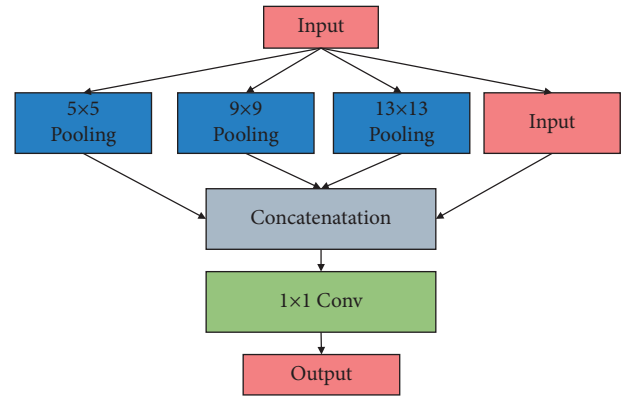

Figure 4: The operation flow of the SPP. SPP pools the final output of CSPDarkNet at different kernels and concatenates the results to produce a high-dimensional feature map.

the diagram of the CAM, where C denotes the input's channel, while H and W, respectively, refer to its height and width.

Next, the outputs from the backbone (Dark2, Dark3, and Dark4) are fed into the CAM to increase the response of bolts. Subsequently, the path aggregation feature pyramid network (PAFPN) [21] processes the outputs from the CAM. Feature pyramid network (FPN) transfers semantic features top-down, and PAN transfers positioning features bottom-up. PAFPN combines them for a better feature fusion and directly outputs the multiscale feature maps. The decoupled detection head of YOLOx utilizes these feature maps to predict the position of bolts. Lastly, the detection branch outputs the location information of the bolts.

It is worth noting that the backbone and bolt detection branches constitute YOLOx. YOLOx is a target detection model that uses YOLOv3 as the baseline, with CSPDarkNet53 as the backbone, in which a detector containing decoupled prediction branches is employed [46]. CSPDarkNet53 is used to extract the depth features of the input image from shallow to deep layers, and the detector is used to predict the target location from the above features.

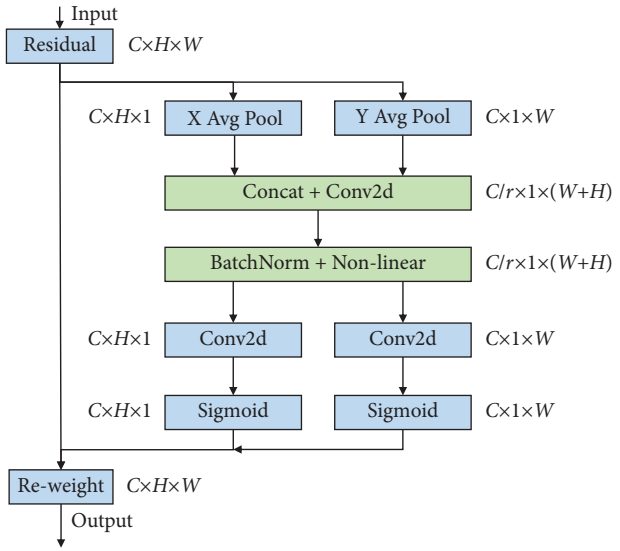

Figure 5: The operation flow of the CAM. CAM incorporates spatial location information into the channel attention mechanism while maintaining a low computational cost.

3.1.3. Corrosion Segmentation. The corrosion on bolts generally has a limited area, sparse distribution, and an unpredictable shape. Therefore, it is crucial to utilize the information on the resolution when performing the segmentation of the corrosion area. The structure of the segmentation branch when the size of the input is 640×640 is illustrated in Figure 6.

However, the down-sampling operation in the backbone results in a loss of resolution information in the image. In order to effectively utilize the available resolution information, the decoder in the segmentation branch is equipped with multiple skip-connection structures. The skip-connection structure incorporates features from the backbone (Dark1, Dark2, Dark3, and Dark4) into the decoder by concatenating them with the decoder's feature maps. This augmentation facilitates the recovery of lost resolution information, consequently bolstering the performance of the decoder.

The integration of feature maps from the backbone and the decoder is expedited by implementing a skip-connection structure, which concatenates the feature maps obtained from the backbone with those generated by the decoder. As depicted in Figure 6, after the concatenation and upsampling operation, it is necessary to process the resultant fused feature effectively. Consequently, we incorporate the CSP layer into the design to facilitate the extraction of features from the concatenated feature map. As outlined in Section 3.1.1, the CSP layer demonstrates the ability to improve learning performance while concurrently reducing computational overhead. Furthermore, it contributes to the amplification of the response from the corrosion area in both the backbone and the preceding layers of the decoder.

The decoder generates a confidence map with dimensions equivalent to that of the original image. Each pixel in the final confidence map represents the probability of the

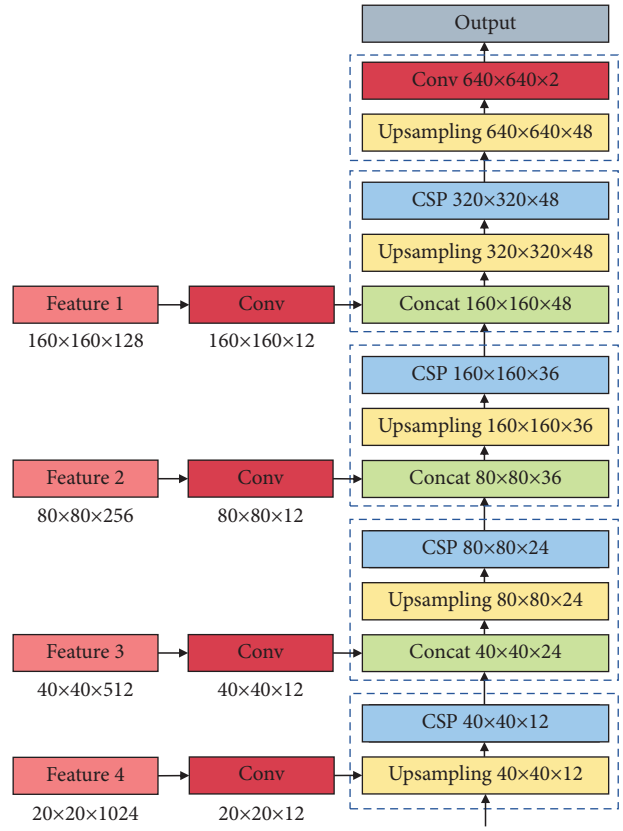

Figure 6: The structure of the CSP-based decoder. The decoder upsamples the feature while fusing features from the output ends of the backbone to recover the resolution information. Lastly, it gives a mask representing the location of the corrosion area.

pixel belonging to either the background or the corrosion area. The binary determination of a pixel affiliation with the target area is determined using a preestablished threshold value of 0.5. If the value of a specific pixel surpasses this threshold, it is set to 1 to denote its inclusion in the corrosion area; conversely, if it falls below the threshold, it is set to 0 to signify its presence in the background. Consequently, this process yields a mask that delineates the corrosion area.

Our strategy combines the strength of lightweight feature extraction with robust capabilities by incorporating the lightweight CSPDarkNet architecture into the backbone network. Furthermore, we amplify YOLOX performance by integrating CAM into the detection branch, and in the segmentation branch, we formulate a decoder based on the CSP methodology. These architectural choices collectively optimize the overall model performance.

3.2. Multitask Learning. The upper and lower parts of Figure 1 share some features output by the backbone, and each independently further processes the features and completes the tasks of bolt detection and corrosion area segmentation. As illustrated in Figure 7, the detection branch in the proposed model labels both healthy and corroded bolts, while the segmentation branch is responsible for identifying and segmenting the corroded areas. The

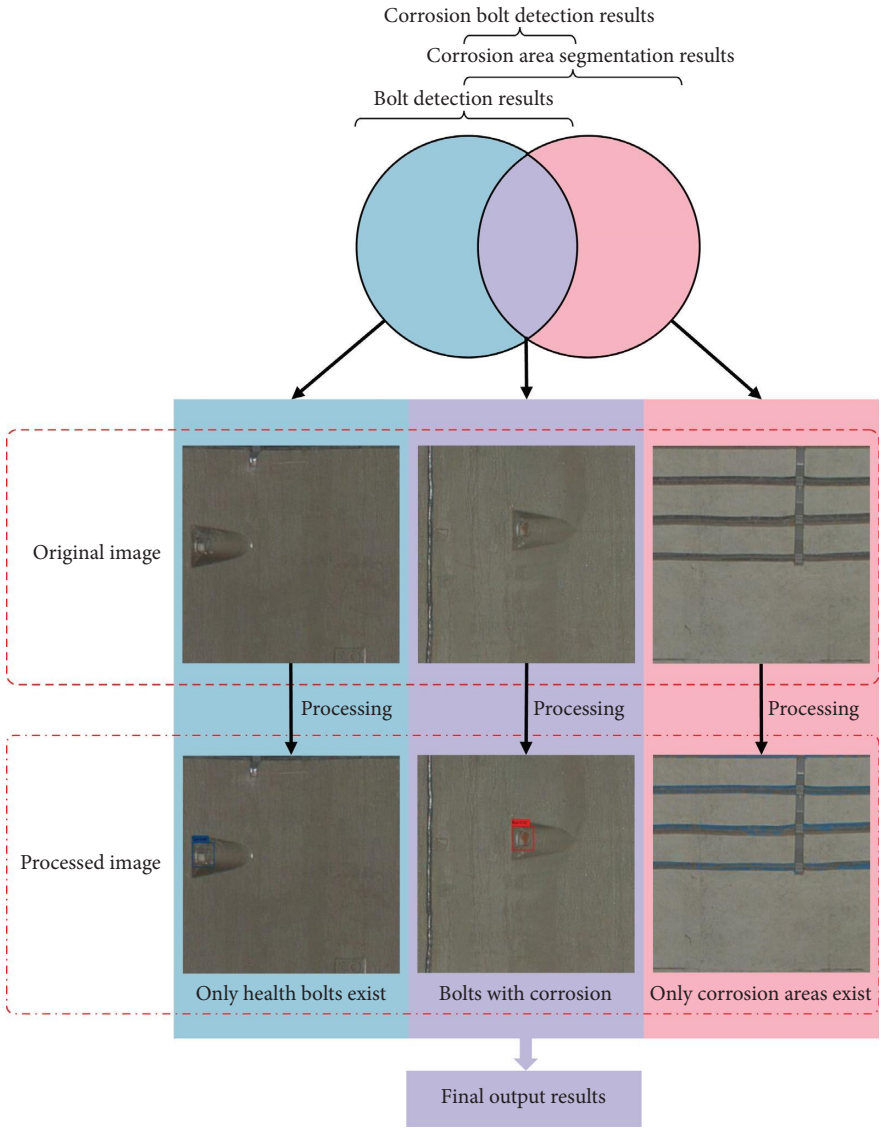

FIGURE 7: Multitask learning approach limits the segmentation region. The pink part represents the bolt detection branch, while the blue part shows the corrosion segmentation branch. The bolt corrosion area is determined by intersecting the two parallel branches.

collaboration between the two modules enables the segmentation to focus solely on the corrosion of the bolts. Although connecting two parts in series can also restrict the region to corroded bolts, it requires two backbones, resulting in significantly more computational time. Additionally, the two backbones cannot share parameters during training and evaluation, which necessitates additional resources (such as datasets, time, and computing power) for training compared with the dual structure proposed in this study.

The detection loss, represented by L_{det}, is a combination of three loss terms, i.e., the classification loss, object loss, and bounding box loss, which is weighted and combined as described in

$$L_{\text{det}} = a_1 L_{\text{class}} + a_2 L_{\text{obj}} + a_3 L_{\text{box}}, \tag{1}$$

where L_{class} and L_{obj} are focal loss [47], and a_1, a_2, and a_3 can be tuned to balance all parts of the detection loss. They are utilized to reduce the loss of well-classified examples, thus forcing the network to focus on the hard ones. L_{class} is used for classifying different kinds of objects, and L_{obj} is used for classifying the object and the background. L_{box} is L_{CIoU} [48], which takes distance, overlap rate, the similarity of scale, and aspect ratio between the predicted box and GT into consideration.

The segmentation loss L_{seg} contains L_{BCE} and L_{Dice} [49]. Binary cross entropy (BCE) loss aims at minimizing the classification errors between pixels of network outputs and the targets, and the dice loss has been widely used in small target segmentation. L_{BCE} and L_{Dice} are defined as follows:

$$L_{\mathrm{BCE}} = y \log p + (1 - y)\log(1 - p),$$

$$L_{\mathrm{Dice}} = 1 - \frac{I + \varepsilon}{U - I + \varepsilon},$$

$$I = \sum_{1}^{N} p_i y_i, \tag{2}$$

$$U = \sum_{1}^{N} (t_i + y_i),$$

where y represents targets, p denotes the prediction probability output by the network, and N expresses the item of the targets.

The segmentation loss is shown as

$$L_{\mathrm{seg}} = L_{\mathrm{BCE}} + L_{\mathrm{Dice}}. \tag{3}$$

In brief, the final loss is a weighted sum of the two parts:

$$L_{\mathrm{all}} = b_1 L_{\mathrm{det}} + b_2 L_{\mathrm{seg}}, \tag{4}$$

where b_1 and b_2 can be tuned to balance all parts of the total loss.

In this section, we propose a multitask strategy that logically processes shared backbone features independently for both bolt detection and corrosion area segmentation. This innovative approach fosters a seamless collaboration between the detection and segmentation branches, ultimately resulting in a logical reduction of both the missed detection rate (MDR) and false alarm rate (FAR), thereby enhancing accuracy and practicality.

3.3. TensorRT Speedup. Given the practical application, it is imperative for the model to exhibit a fast inference speed to conform to the deployment requirements. Thus, the model should be optimized to enhance its performance and support its development.

In this study, TensorRT, a deep learning optimizer developed by NVIDIA, is adopted to increase the inference speed of the proposed model. TensorRT exhibits the prominent capability of providing low-latency and high-throughput deployment inference for deep learning models, thus taking on critical significance in conforming to the deployment requirements in practical applications. It supports the use of three types of computation, including kFLOAT (float32), kHALF (float16), and kINT8 (int8), which allows for acceleration using low-precision data types. The network structure is optimized in TensorRT by combining similar operations, thus simplifying the computation. Additionally, it also optimizes the usage of video memory and GPU bandwidth, depending on the framework and GPU used. Furthermore, it significantly increases the inference speed of the model in real-world applications.

4. Experimental Settings and Results Analysis

This section provides a comprehensive evaluation of the DSNet proposed in this study using the collected dataset. We conduct a comparative analysis between our model and state-of-the-art methods that focus on single tasks, assessing both quantitative performance and inference speed. This comparison illustrates the overall superiority of our model. Additionally, we perform ablation studies to analyze the impact of each component and experimental setup in our model. Considering the complexity of the model and the extensive computations involved, the network parameters are set empirically.

4.1. Data Acquisition System and Dataset. Figure 8 shows the data acquisition system named MS100 which is produced by South Surveying and Mapping Technology Co., Ltd.

The MS100 can automatically move and scan the tunnel panorama at a speed of 1 km/h in disease-scanning mode. The images acquired by the MS100 are first corrected to orthophoto through its orthography correction. After orthography correction is completed, the acquired image resolution reaches 2 mm at a distance of 5 meters, which satisfies the actual demand for water leakage identification. The specific parameters of the MS100 scanner are listed in Table 2. Our experiments were performed on the corroded bolt dataset collected by the MS100 from a Beijing metro tunnel in service. The corroded bolt dataset consists of 1441 pictures in the size of 640 × 640. We labeled the bolts with GT boxes (the blue boxes in Figure9(d)) and the corrosion area of the bolts with GT masks (the yellow area in Figure 9(d)). The data were labeled in a VOC format. The experiment randomly selects 287 images as the test set. Our experiments show the performance of the model on the test set. The dataset can be accessed at https://github.com/StreamHXX/Tunnel-lining-disease-image.

4.2. Evaluation Metrics. In this paper, the performance of different models will be evaluated by precision rate, recall rate, F1 score, average precision (AP), mean intersection over union (mIoU), mean pixel accuracy (mPA), accuracy, frames per second (FPS), and parameters.

For detection performance, precision rate, recall rate, and F1 score are defined as

$$\mathrm{Precision} = \frac{X_{\mathrm{TP}}}{X_{\mathrm{TP}} + X_{\mathrm{FP}}},$$

$$\mathrm{Recall} = \frac{X_{\mathrm{TP}}}{X_{\mathrm{TP}} + X_{\mathrm{FN}}}, \tag{5}$$

$$\mathrm{F1\ score} = \frac{2 \times \mathrm{precision} \times \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}},$$

where X_{TP} denotes the number of objects identified as true and X_{FP} denotes the number of objects identified as false. AP represents the area under the precision-recall (P-R) curve. The higher the above-mentioned metrics, the better the detection performance will be.

For segmentation performance, IoU represents the ratio of intersection and union between the GT area and the predicted segmentation area. As shown in Figure 10, the

Figure 8: A picture of the MS100.

Table 2: Specific parameters of the MS100 scanner.

Classes	Parameters
Operating mode	Phase
Scanning distance	0.6–350 meters
Range accuracy	±1 millimeter
Angular accuracy	≥19 seconds of arc
Scanning speed	1,000,000 points per second
Camera	Coaxial image, built-in 165 million pixels

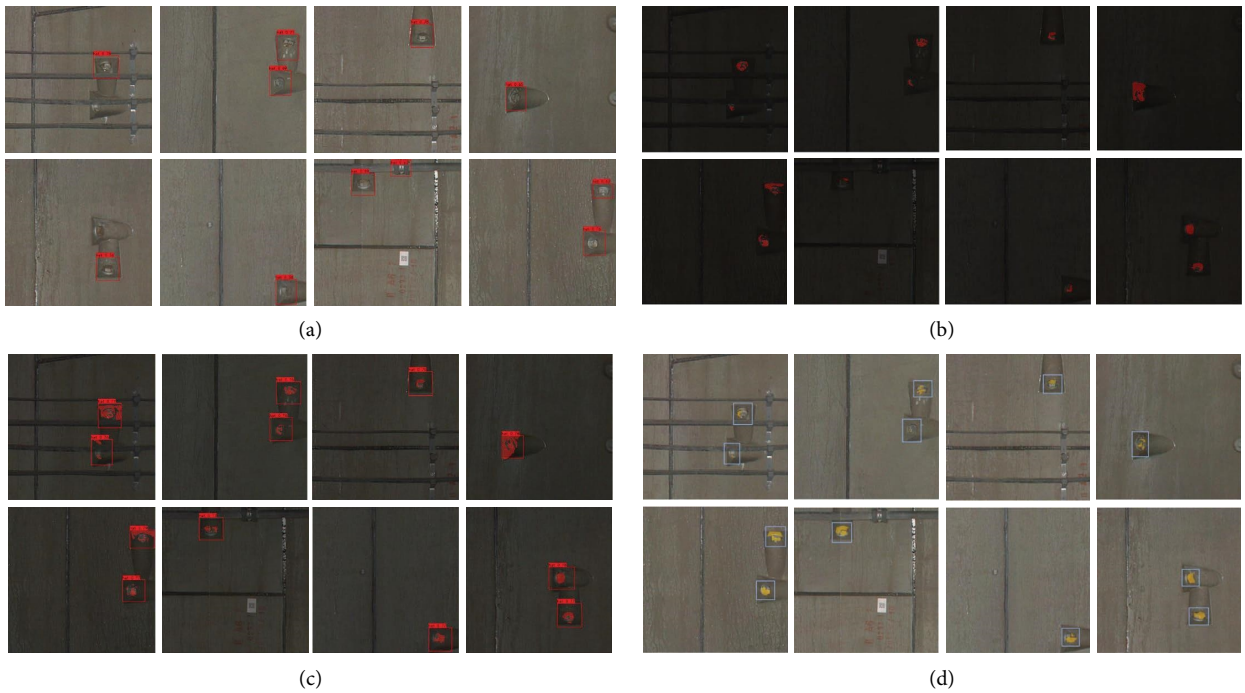

Figure 9: Some examples of results. (a) EfficientDet. (b) HRNet. (c) DSNet. (d) GT.

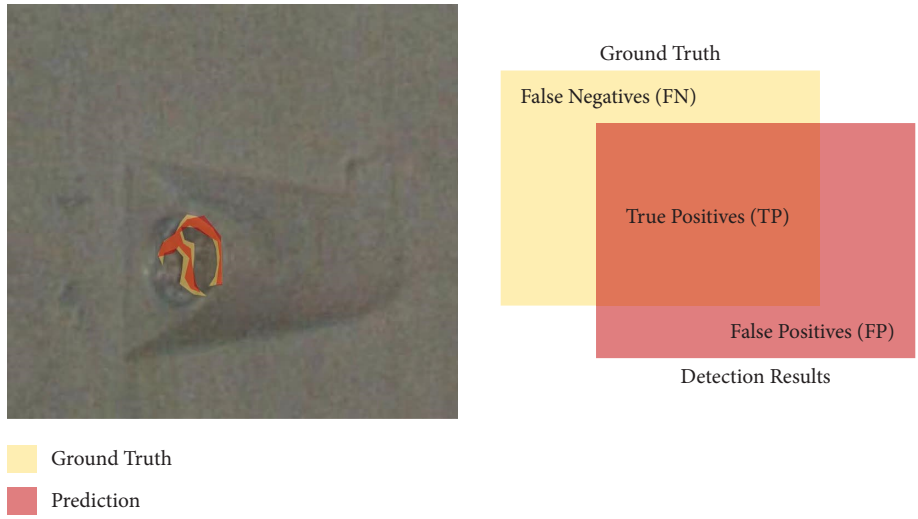

Figure 10: The definition of IoU.

yellow part in the image represents the ground truth area A_1, and the red part in the image represents the prediction area A_2. We record the intersection of the two areas as A_3. The IoU in segmentation can be calculated as

$$\text{IoU} = \frac{A_3}{A_1 + A_2 - A_3}. \tag{6}$$

The mIoU used in this study is the average value of IoU of all classes. Pixel accuracy (PA) indicates the accuracy of classifying each pixel in the image in each class. The mPA represents the average value of PA of all classes. Accuracy measures the average classification accuracy of an image for each pixel. Higher these metrics indicate better segmentation performance.

As for speed performance, FPS is the number of image frames that the model can process per second. FPS shows the speed and complexity of different models. Parameters represent the size of memory space occupied by model parameters. Higher FPS indicates better speed performance. Lower parameters indicate less space occupied, which is better for deployment.

Besides, in order to evaluate the performance of our approach on the corroded bolt detection task level, we employ the MDR and FAR as evaluation metrics in our experiments. MDR and FAR can be calculated as

$$\begin{aligned} \text{MDR} &= \frac{\text{FN}}{\text{FN} + \text{TP}}, \\ \text{FAR} &= \frac{\text{FP}}{\text{TN} + \text{FP}}, \end{aligned} \tag{7}$$

where true positive (TP) and true negative (TN), respectively, denote the number of positive samples and negative instances correctly detected, respectively; false positive (FP) and false negative (FN) express the number of negative cases and positive samples wrongly detected, respectively. TP, TN, FP, and FN are computed after the intersection operation since they express the performance on the corroded bolt detection task level.

4.3. Implementation Details

4.3.1. Experiment Settings. The experiments are all implemented on an Intel® Core™ i7-11700K CPU (3.6 GHz, 3.2 GB RAM) and an NVIDIA GeForce RTX 3060 GPU (CUDA version 11.6) with Python 3.9.12 (PyTorch 1.11.0) in 64 Bit Ubuntu 18.04.1 Long Term Support operating system.

The experiment sets the resolution of the input image to 640×640. We use the stochastic gradient descent (SGD) with 0.9 momenta as the optimizer to find the optimal parameters and initialize the learning rate to 0.001 in the training process. We chose the cosine delay with a warm-up as the learning rate schedule. All models have been fully trained in an end-to-end way. All experiments follow the same experimental settings and evaluation metrics.

4.3.2. Image Augmentation. Due to the limited field data, the data collected from the actual tunnel environment were augmented through rotation, shear, translation, and mosaic techniques, which are illustrated in Figure 11. We utilized a rotation rate of 0.5 and a translation rate of 0.1 during image rotation and translation procedures. The image-cutting process incorporated a scale rate and a shear rate of 0.5. Furthermore, we employed mosaic in our experiments to augment the performance of the decoder, using a mosaic rate of 1.0. This technique significantly boosted dataset background diversity by seamlessly combining multiple cropped images. Additionally, the augmented dataset is used uniformly for training the comparison models and ablation models.

4.3.3. Transfer Learning. With the limited data collected, it is hard to fully train a new model from scratch. To address this issue, a common approach is to leverage a pretrained model that has already been trained on larger datasets. In this study, we utilize a pretrained CSPDarkNet as the backbone of our model and fine-tune it to adapt to the task of bolt detection and corrosion area segmentation. The use of a pretrained

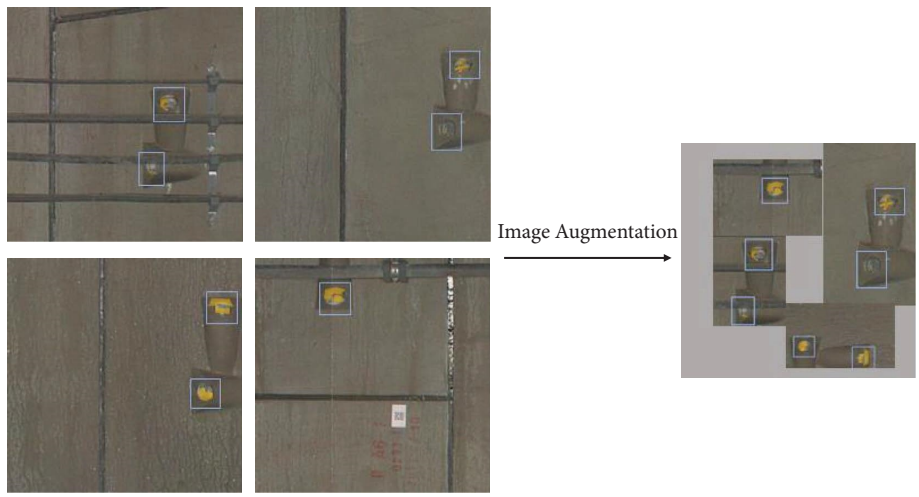

Figure 11: The image augmentation using rotation, shear, translation, and mosaic techniques.

model not only speeds up the convergence process but also leads to a significant improvement in performance compared with models trained from scratch.

4.4. Comparisons on the Dataset. In this section, several advanced methods of segmentation and detection are selected for comparison and fully trained on the dataset of this study. Since two tasks (i.e., detection and segmentation) exist in this work, we use the comprehensive results of the model and compare the performance of the two tasks, respectively. As indicated by the results, the proposed DSNet outperforms other methods for comparison in detection and segmentation.

For detection, we compare and evaluate the performance of the proposed model and the comparison methods on the test set in terms of accuracy and speed. Table 3 lists the comparison results. In Table 3, faster R-CNN [50] acts as a two-stage CNN-based object detector, i.e., a typical non-end-to-end model. EfficientDet [51] is a detection model proposed by Google following the classification model EfficientNet, where the bidirectional FPN is incorporated for characteristic formaldehyde fusion. YOLOv5s is a light-weight end-to-end model with high accuracy and speed. The current mainstream object detection models are selected for performance comparison.

As shown in Table 3, compared with faster R-CNN, EfficientDet, and YOLOv5s, our proposed DSNet has achieved the best recall rate and got a 0.022, 0.064, and 0.025 better in recall, respectively. It shows that the method proposed rarely fails to detect bolts. For the F1 score, DSNet and YOLOv5s perform better than faster R-CNN and EfficientDet. Both of the two models achieve 0.960. From the perspective of AP, DSNet gets 0.972 AP@0.5 and 0.471 AP@0.5: 0.95, which is the optimal among all models in Table 3. The above results indicate that DSNet has the best performance in detection accuracy, especially in AP@0.5: 0.95, which confirms the effectiveness of DSNet in the detection task.

In terms of the size and speed of the model, we can see that DSNet has no obvious advantage in parameters and FPS. However, it should be noted that the DSNet proposed has two branches to resolve two tasks. The comparison methods in Table 3 only focus on the detection task. Despite this, DSNet is still much lighter than faster R-CNN in parameters and faster than faster R-CNN and EfficientDet.

We also evaluate and compare our proposed DSNet with several advanced models in segmentation in terms of accuracy and speed, as shown in Table 4. U-Net [24] is an end-to-end segmentation model widely used in the medical image field. Under a small amount of data, U-Net can guarantee high accuracy. Deeplab v3+ [52] is an optimized version of the Deeplab series models, which introduces an encoder-decoder structure to increase the edge segmentation accuracy. Unlike U-Net, HRNet [20] achieves strong semantic information and accurate spatial information utilizing parallel different resolution branches, thus avoiding loss of information in down-sampling. Compared with the above methods, we illustrate the performance of our model more clearly.

From the numerical results in Table 4, U-Net, Deeplab v3+, HRNet, and DSNet achieve the same accuracy as high as 0.997 in the corrosion area segmentation task. For mIoU, HRNet outperforms other models with 0.685. Our proposed model DSNet also achieves a near-performance (0.682). Moreover, DSNet gets 0.786 on mPA, which is the best among all comparison methods.

Taking the requirements of the engineering application into consideration, both size and speed are crucial indices in the segmentation task. As presented in Table 4, the state-of-the-art segmentation method, Deeplab v3+, attains optimal performance in terms of both speed and size. Despite being a dual-task model, the proposed DSNet still achieves a frame rate of 25.82 FPS. With acceleration from TensorRT, the frame rate of DSNet increases to 34.01 FPS, representing a significant improvement over U-Net and HRNet, with frame rate increases of 22.37 FPS and 20.57 FPS, respectively.

Table 3: Detection results.

Model	Precision	Recall	F1 score	AP@0.5	AP@0.5: 0.95	Parameters (M)	FPS
Faster R-CNN	0.899	0.959	0.930	0.924	0.337	108.2	11.95
EfficientDet	**0.968**	0.917	0.940	0.954	0.440	**25.6**	14.22
YOLOv5s	0.959	0.956	**0.960**	0.948	0.435	27.1	**53.50**
DSNet (ours)	0.943	**0.981**	**0.960**	**0.972**	**0.471**	34.5	25.82
DSNet (TensorRT)						73.9	34.01

The bold values are the best value among all comparents. Illustrating that DSNet is better than other detection-only methods.

Table 4: Segmentation results.

Model	mIoU	mPA	Accuracy	Parameters (M)	FPS
U-Net	0.680	0.769	0.997	95.0	11.64
Deeplab v3+	0.670	0.751	0.997	**25.3**	**36.93**
HRNet	0.685	0.782	0.997	37.5	13.44
DSNet (ours)	0.682	**0.786**	0.997	35.4	25.82
DSNet (TensorRT)				73.9	34.01

The bold values are the best value among all comparents.

As shown in Table 5, we have combined the above detection task models and segmentation task models with each other for experiments and compared the schemes for detecting corroded bolts directly using fast R-CNN, YOLOv5s, YOLOv5n6, YOLOv5n, and ensemble YOLOv5n [2]. Our proposed DSNet achieved excellent results in terms of MDR and FAR. Accurately, the calculated MDR was only 0.019, and the FAR was only 0.017, respectively, which was the best performance among all the compared combinations. Moreover, the complexity of our multitask model compared to other models is also reflected in Table 5. GFLOPs (giga floating point of operations) are the number of floating point operations, which can be used to measure model complexity. In Table 5, the GFLOPs of different models are calculated and compared. We can see that our proposed DSNet is smaller than any combination of segmentation models and detection models.

Figure 11 shows some examples of results from different models. As for detection, compared with the EfficientDet, the DSNet gives bolts detected higher confidence. Additionally, the DSNet outperforms the EfficientDet in terms of the detection rate for bolts. EfficientDet fails to detect a bolt in the first image of Figure 9. As for segmentation, compared with the HRNet, the proposed DSNet successfully labels some areas of corrosion that were not identified by the HRNet. However, it also leads to the incorrect segmentation of some areas.

In general, the DSNet exhibits better comprehensive performance in bolt detection and corrosion area segmentation.

4.5. Ablation Study. In the present section, a comprehensive examination of the ablation studies is presented. The ablation experiments fall into two parts as follows. The first part refers to the loss function employed in the segmentation task, while the other part involves a variety of components of the DSNet. Subsequently, the numerical experimental results are analyzed in depth.

4.5.1. Ablation Study of Transfer Learning. As shown in Table 6, we validate the effect of transfer learning in our experiment. Using transfer learning, the DSNet proposed gets higher results in AP@0.5, AP@0.5: 0.95, mIoU, and mPA. The precision and F1 score of the model with transfer learning are a little lower. However, we can see that transfer learning enhances the comprehensive performance of the model in both bolt detection and corrosion area segmentation. Moreover, transfer learning also accelerates the convergence speed of the model in training [53]. The experimental result confirms that it can make the DSNet better in our dataset.

4.5.2. Ablation Study of Loss Function. As shown in Table 7, we perform an ablation study on the test set to verify the effect of different combinations of L_{seg}. The combination of BCE and dice loss exhibits the optimal performance. Compared with the focal and dice loss, the combination of BCE and dice loss outperforms all the metrics. The above combination of L_{seg} not only enhances the performance of corrosion area segmentation but also makes the task of bolt detection better. As revealed by the ablation study of the loss function, the combination of BCE and dice loss achieves the optimal performance in this study.

4.5.3. Ablation Study of Components in DSNet. As shown in Table 8, we also verify the impact of two components in our model on the test set. Table 8 represents the CSP component of the segmentation decoder. The baseline is the DSNet without CAM and CSP. With CAM, it can be seen that all indicators of detection have a significant improvement compared to the baseline, which proves the effectiveness of CAM on the task of bolt detection. Compared to the DSNet with CAM, DSNet with CAM and CSP achieves better performance at indicators of corrosion area segmentation task. mIoU and mPA are improved from 0.665 and 0.727 to 0.682 and 0.786, respectively. Although some indicators of the bolts detection task (precision, recall, F1 score, and AP@0.5) decline, the more comprehensive indicators of bolts detection, AP@0.5: 0.95 increases from 0.454 to 0.471. Comprehensively, the model with both CAM and CSP makes a better performance, which verifies the necessity and rationality of each component in our model.

Table 5: Comparisons of combined approaches.

Detection task			Segmentation task			MDR	FAR	FPS	GFLOPs
Faster R-CNN	EfficientDet	YOLOv5s	U-Net	Deeplab v3+	HRNet				
√			√			0.039	0.093	5.89	553.726
√				√		0.044	0.097	9.03	249.003
√					√	0.033	0.089	6.33	230.013
	√		√			0.088	0.032	6.41	356.574
	√			√		0.086	0.029	10.27	51.851
	√				√	0.074	0.035	6.91	32.861
		√	√			0.049	0.042	9.56	368.869
		√		√		0.041	0.051	21.83	64.146
		√			√	0.036	0.039	10.74	45.156
		Faster R-CNN [30]				0.083	0.31	11.91	200.857
		YOLOv5s [2, 36]				0.038	0.129	156.37	16.000
		YOLOv5n6 [2, 36]				0.067	0.123	85.31	4.300
		YOLOv5n [2, 36]				0.038	0.111	150.56	**4.200**
		Ensemble YOLOv5n [2]				0.03	0.088	141.64	12.600
		DSNet (ours)				**0.019**	**0.017**	25.82	13.968
		DSNet (TensorRT)						**34.01**	10.832

The bold values are the best value among all comparents. Illustrating that our method is better than other approaches.

Table 6: Ablation study (transfer learning).

Transfer learning	Precision	Recall	F1 score	AP@0.5	AP@0.5: 0.95	mIoU	mPA
	0.957	0.981	**0.970**	0.968	0.462	0.679	0.769
√	0.943	0.981	0.960	**0.971**	**0.471**	**0.682**	**0.786**

The bold values are the best value among all comparents. Illustrating that transfer learning is better on many targets.

Table 7: Ablation study (loss function).

Loss Function			Precision	Recall	F1 score	AP@0.5	AP@0.5: 0.95	mIoU	mPA
BCE	Focal	Dice							
	√	√	0.940	0.980	0.950	0.967	0.462	0.642	0.675
√		√	**0.943**	**0.981**	**0.960**	**0.971**	**0.471**	**0.682**	**0.786**

The bold values are the model with BCE and Dice which gets better performance than that with Focal and Dice at diverse metrics.

Table 8: Ablation study (component).

Module			Precision	Recall	F1 score	AP@0.5	AP@0.5: 0.95	mIoU	mPA
Base	CAM	CSP							
√			0.951	0.917	0.930	0.934	0.405	0.623	0.664
√	√		**0.954**	**0.986**	**0.970**	**0.979**	0.454	0.665	0.727
√	√	√	0.943	0.981	0.960	0.971	**0.471**	**0.682**	**0.786**

The bold values are the best value among all comparents.

5. Discussion

The results of the above-mentioned experiments have shown the effectiveness of the proposed DSNet. It should be noted that DSNet has not achieved the optimal speed performance in both the detection and segmentation experiments because DSNet needs to handle the two tasks of bolt detection and corrosion area segmentation. In the experiment, the operation mode of the code lets the DSNet finish the bolts detection task first and then the corrosion area segmentation task. However, we can see from the comparisons that the inference speed of the DSNet is only a little different from the best speed performance of the segmentation task, although the DSNet needs to perform two tasks. Thus, we believe that by using multithreading technology to handle these two tasks simultaneously, the DSNet can get similar performance with the models focusing on a single task.

6. Conclusion

In this paper, we propose an end-to-end DSNet to detect bolts and segment corrosion simultaneously. The detection branch incorporates the CAM module, which amplifies the focus of the model on the bolts, and the segmentation branch utilizes a designed decoder based on the CSP module to generate a mask for identifying the corroded regions of bolts.

The experiments and ablation studies show that our DSNet outperforms other methods and the components perform well in evaluation. It achieves a precision of 0.957 and a recall of 0.981 under transfer learning for the detection task while reaching a mIoU of 0.682 and an mPA of 0.786 for corrosion segmentation. The frame rate of our proposed DSNet can reach 34.01 FPS with TensorRT speedup. Although there is a decline compared with YOLOv5s, the overall MDR of the method was as low as 0.019 and the overall FAR was as low as 0.017, which demonstrated that it can significantly reduce the occurrence of missed detection and false alarm. We also believe that by using multithreading and distributed computing to develop the model, it can achieve similar speed performance with the optimal methods for comparison.

Data Availability

The data are publicly available at https://github.com/StreamHXX/Tunnel-lining-disease-image.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grant 62120106011, Grant 52172323, and Grant U22A2046 and the Technical Innovation Project of Beijing Municipal Road and Bridge Co., Ltd. (Science and Technology Company-Scientific Research-W-22064).

References

[1] S. M. Y. Nikravesh and M. Goudarzi, "A review paper on looseness detection methods in bolted structures," *Latin American Journal of Solids and Structures*, vol. 14, no. 12, pp. 2153–2176, 2017.

[2] L. Tan, T. Tang, and D. Yuan, "An ensemble learning aided computer vision method with advanced color enhancement for corroded bolt detection in tunnels," *Sensors*, vol. 22, no. 24, p. 9715, 2022.

[3] L. Tan, X. Hu, T. Tang, and D. Yuan, "A lightweight metro tunnel water leakage identification algorithm via machine vision," *Engineering Failure Analysis*, vol. 150, Article ID 107327, 2023.

[4] J. Yang and F.-K. Chang, "Detection of bolt loosening in c–c composite thermal protection panels: I. diagnostic principle," *Smart Materials and Structures*, vol. 15, no. 2, pp. 581–590, 2006.

[5] Q.-B. Ta, T.-C. Huynh, Q.-Q. Pham, and J.-T. Kim, "Corroded bolt identification using mask region-based deep learning trained on synthesized data," *Sensors*, vol. 22, no. 9, p. 3340, 2022.

[6] X. Hu, Y. Cao, T. Tang, and Y. Sun, "Data-driven technology of fault diagnosis in railway point machines: review and challenges," *Transportation Safety and Environment*, vol. 4, no. 4, 2022.

[7] X. Hu, Y. Cao, Y. Sun, and T. Tang, "Railway automatic switch stationary contacts wear detection under few-shot occasions," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 9, pp. 14893–14907, 2022.

[8] Y. Pan, Y. Ma, Y. Dong, Z. Gu, and D. Wang, "A vision-based monitoring method for the looseness of high-strength bolt," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–14, 2021.

[9] Z. Zhao, H. Qi, Y. Qi, K. Zhang, Y. Zhai, and W. Zhao, "Detection method based on automatic visual shape clustering for pin-missing defect in transmission lines," *IEEE Transactions on Instrumentation and Measurement*, vol. 69, no. 9, pp. 6080–6091, 2020.

[10] J. Zhong, Z. Liu, H. Wang et al., "A looseness detection method for railway catenary fasteners based on reinforcement learning refined localization," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–13, 2021.

[11] L. Xiao, B. Wu, and Y. Hu, "Missing small fastener detection using deep learning," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–9, 2021.

[12] Q. Zhou, Z. Qu, Y.-X. Li, and F.-R. Ju, "Tunnel crack detection with linear seam based on mixed attention and multiscale feature fusion," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–11, 2022.

[13] S. Lee, L.-M. Chang, and M. Skibniewski, "Automated recognition of surface defects using digital color image processing," *Automation in Construction*, vol. 15, no. 4, pp. 540–549, 2006.

[14] S. Livens, P. Scheunders, G. Van de Wouwer et al., "Classification of corrosion images by wavelet signatures and lvq networks," in *Proceedings of the Computer Analysis of Images and Patterns: 6th International Conference*, pp. 538–543, Prague, Czech Republic, September 1995.

[15] S. K. Fondevik, A. Stahl, A. A. Transeth, and O. Ø. Knudsen, "Image segmentation of corrosion damages in industrial inspections," in *Proceedings of the 2020 IEEE 32nd International Conference on Tools with Artificial Intelligence (ICTAI)*, pp. 787–792, IEEE, Baltimore, MD, USA, November 2020.

[16] Q. Hou, D. Zhou, and J. Feng, "Coordinate attention for efficient mobile network design," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 713–722, Nashville, TN, USA, June 2021.

[17] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "CBAM: convolutional block attention module," 2018, https://arxiv.org/abs/1807.06521.

[18] C.-Y. Wang, H.-Y. M. Liao, I.-H. Yeh, Y.-H. Wu, P.-Y. Chen, and J.-W. Hsieh, "CSPNet: a new backbone that can enhance learning capability of CNN," 2019, https://arxiv.org/abs/1911.11929.

[19] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3431–3440, Boston, MA, USA, June 2015.

[20] K. Sun, B. Xiao, D. Liu, and J. Wang, "Deep high-resolution representation learning for human pose estimation," 2019, https://arxiv.org/abs/1902.09212.

[21] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, "Path aggregation network for instance segmentation," 2018, https://arxiv.org/abs/1803.01534.

[22] Y.-J. Cha, W. Choi, G. Suh, S. Mahmoudkhani, and O. Büyüköztürk, "¨ "Autonomous structural visual inspection using region-based deep learning for detecting multiple damage types," *Computer-Aided Civil and Infrastructure Engineering*, vol. 33, no. 9, pp. 731–747, 2018.

[23] A. Bochkovskiy, C. Wang, and H. M. Liao, "Yolov4: optimal speed and accuracy of object detection," 2020, https://arxiv.org/abs/2004.10934.

[24] O. Ronneberger, P. Fischer, and T. Brox, "U-net: convolutional networks for biomedical image segmentation," 2015, https://arxiv.org/abs/1505.04597.

[25] C. Wang, N. Wang, S.-C. Ho, X. Chen, and G. Song, "Design of a new vision-based method for the bolts looseness detection in flange connections," *IEEE Transactions on Industrial Electronics*, vol. 67, no. 2, pp. 1366–1375, 2020.

[26] A. R. M. Forkan, Y.-B. Kang, P. P. Jayaraman et al., "Corr-detector: a framework for structural corrosion detection from drone images using ensemble deep learning," *Expert Systems with Applications*, vol. 193, Article ID 116461, 2022.

[27] R. Zhang, S. Chang, Z. Wei, Y. Zhang, S. Huang, and Z. Feng, "Modulation classification of active attacks in internet of things: lightweight mcbldn with spatial transformer network," *IEEE Internet of Things Journal*, vol. 9, no. 19, Article ID 19132, 19146 pages, 2022.

[28] T.-C. Huynh, J.-H. Park, H.-J. Jung, and J.-T. Kim, "Quasi-autonomous bolt-loosening detection method using vision-based deep learning and image processing," *Automation in Construction*, vol. 105, Article ID 102844, 2019.

[29] Y. Xu, D. Li, Q. Xie, Q. Wu, and J. Wang, "Automatic defect detection and segmentation of tunnel surface using modified mask r-cnn," *Measurement*, vol. 178, Article ID 109316, 2021.

[30] T.-C. Huynh, "Vision-based autonomous bolt-looseness detection method for splice connections: design, lab-scale evaluation, and field application," *Automation in Construction*, vol. 124, Article ID 103591, 2021.

[31] H. Gong, X. Deng, J. Liu, and J. Huang, "Quantitative loosening detection of threaded fasteners using vision-based deep learning and geometric imaging theory," *Automation in Construction*, vol. 133, Article ID 104009, 2022.

[32] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: unified, real-time object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 779–788, Honolulu, HI, USA, July 2016.

[33] S. Yanan, Z. Hui, L. Li, and Z. Hang, "Rail surface defect detection method based on yolov3 deep learning networks," in *Proceedings of the 2018 Chinese Automation congress (CAC)*, pp. 1563–1568, IEEE, Xi'an, China, November 2018.

[34] K. Guo, C. He, M. Yang, and S. Wang, "A pavement distresses identification method optimized for yolov5s," *Scientific Reports*, vol. 12, no. 1, p. 3542, 2022.

[35] X. Yang, Y. Gao, C. Fang, Y. Zheng, and W. Wang, "Deep learning-based bolt loosening detection for wind turbine towers," *Structural Control and Health Monitoring*, vol. 29, no. 6, Article ID e2943, 2022.

[36] X. Zhu, S. Lyu, X. Wang, and Q. Zhao, "Tph-yolov5: improved yolov5 based on transformer prediction head for object detection on dronecaptured scenarios," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 2778–2788, Montreal, Canada, October 2021.

[37] A. Benjumea, I. Teeti, F. Cuzzolin, and A. Bradley, "Yolo-z: improving small object detection in yolov5 for autonomous vehicles," 2021, https://arxiv.org/abs/2112.11798.

[38] P. Gunatilake, M. Siegel, A. G. Jordan, and G. W. Podnar, "Image understanding algorithms for remote visual inspection of aircraft surfaces," *Machine Vision Applications in Industrial Inspection V*, vol. 3029, pp. 2–13, 1997.

[39] M. Siegel and P. Gunatilake, "Remote enhanced visual inspection of aircraft by a mobile robot," in *Proceedings of the 1998 IEEE Workshop on Emerging Technologies, Intelligent Measurement and Virtual Systems for Instrumentation and Measurement (ETIMVIS'98)*, pp. 49–58, New York, NY, USA, May 1998.

[40] Z. Zhou, J. Zhang, and C. Gong, "Hybrid semantic segmentation for tunnel lining cracks based on swin transformer and convolutional neural network," *Computer-Aided Civil and Infrastructure Engineering*, vol. 13, 2023.

[41] J. Kim, S. Shim, S.-J. Kang, and G.-C. Cho, "Learning structure for concrete crack detection using robust super-resolution with generative adversarial network," *Structural Control and Health Monitoring*, vol. 2023, Article ID 8850290, 16 pages, 2023.

[42] D. J. Atha and M. R. Jahanshahi, "Evaluation of deep learning approaches based on convolutional neural networks for corrosion detection," *Structural Health Monitoring*, vol. 17, no. 5, pp. 1110–1128, 2018.

[43] C. V. Dung and L. D. Anh, "Autonomous concrete crack detection using deep fully convolutional neural network," *Automation in Construction*, vol. 99, pp. 52–58, 2019.

[44] S.-K. Chen, I. Huang, and P.-H. Chen, "Applying fully convolutional neural networks for corrosion semantic segmentation for steel bridges: the use of u-net," in *Bridge Maintenance, Safety, Management, Life-Cycle Sustainability and Innovations*, pp. 341–346, CRC Press, Boca Raton, FL, USA, 2021.

[45] O. Ronneberger, P. Fischer, and T. Brox, "U-net: convolutional networks for biomedical image segmentation," in *Proceedings of the Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference*, pp. 234–241, Springer, Munich, Germany, October 2015.

[46] Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun, "Yolox: exceeding yolo series in 2021," 2021, https://arxiv.org/abs/2107.08430.

[47] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss ′ for dense object detection," 2017, https://arxiv.org/abs/1708.02002.

[48] Z. Zheng, P. Wang, D. Ren et al., "Enhancing geometric factors in model learning and inference for object detection and instance segmentation," 2020, https://arxiv.org/abs/2005.03572.

[49] X. Li, X. Sun, Y. Meng, J. Liang, F. Wu, and J. Li, "Dice loss for data-imbalanced NLP tasks," 2019, https://arxiv.org/abs/1911.02855.

[50] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," 2015, https://arxiv.org/abs/1506.01497.

[51] M. Tan, R. Pang, and Q. V. Le, "EfficientDet: scalable and efficient object detection," 2019, https://arxiv.org/abs/1911.09070.

[52] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected CRFs," 2014, https://arxiv.org/abs/1412.7062.

[53] K. He, R. Girshick, and P. Dollar, "Rethinking ImageNet pre-training," 2018, https://arxiv.org/abs/1811.08883.