Original Article

Structural Health Monitoring

Structural Health Monitoring
2020, Vol. 19(1) 105–122

DOI: 10.1177/1475921719837509
journals.sagepub.com/home/shm
SAGE

Autonomous bolt loosening detection using deep learning

Yang Zhang[1,2], Xiaowei Sun[3], Kenneth J Loh[4], Wensheng Su[3], Zhigang Xue[3] and Xuefeng Zhao[1,2]

Abstract

Machine vision-based structural health monitoring is gaining popularity due to the rich information one can extract from video and images. However, the extraction of characteristic parameters from images often requires manual intervention, thereby limiting its scalability and effectiveness. In contrast, deep learning overcomes the aforementioned shortcoming in that it can autonomously extract feature parameters (e.g. structural damage) from image datasets. Therefore, this study aims to validate the use of machine vision and deep learning for structural health monitoring by focusing on a particular application of detecting bolt loosening. First, a dataset that contains 300 images was collected. The dataset includes two bolt states, namely, tight and loosened. Second, a faster region-based convolutional neural network was trained and evaluated. The test results showed that the average precision of bolt damage detection is 0.9503. Thereafter, bolts were loosened to various screw heights, and images obtained from different angles, lighting conditions, and vibration conditions were identified separately. The trained model was then employed to validate that bolt loosening could be detected with sufficient accuracy using various types of images. Finally, the trained model was connected with a webcam to realize real-time bolt loosening damage monitoring.

Keywords

Bolt loosening, convolutional neural network, deep learning, imaging, machine vision

Introduction

Aerospace, civil, marine, and mechanical structures can sustain damage any time during their entire operational life cycle. If damage is left undetected, it can propagate to cause component or catastrophic structural failure. Therefore, the aim of any structural health monitoring (SHM) system is to detect early stages of structural damage so that appropriate repairs can be conducted to maintain system performance and functionality. In addition to confirming whether damage exists or not, the goal is to characterize severity, identify the location, and differentiate the types of damage in the system.

Conventional SHM methods mainly rely on measuring structural response due to ambient or forced excitations, typically using sensors such as accelerometers, strain gauges, displacement meters, and inclinometers. Then, modal analysis is used to determine the degree of structural damage,[1] such as by using dynamic fingerprint analysis, system identification, and neural networks.[2] These methods are all based on characterizing the dynamic properties of the structure, which is highly dependent on environmental effects and insensitive to local damage.[3] Furthermore, these detection methods are only suitable for simple (or idealistic) structures. Although the deployment of dense sensor networks (e.g. wireless sensors) could improve structural monitoring performance, structural dynamic properties do not change significantly due to highly localized damage features (e.g. bolt loosening or fatigue crack).[4,5]

[1]School of Civil Engineering, Dalian University of Technology, Dalian, China
[2]State Key Laboratory of Coastal and Offshore Engineering, Dalian University of Technology, Dalian, China
[3]Jiangsu Province Special Equipment Safety Supervision Inspection Institute, Branch of Wuxi, Wuxi, China
[4]Department of Structural Engineering, University of California–San Diego, San Diego, CA, USA

Corresponding author:
Xuefeng Zhao, School of Civil Engineering, Dalian University of Technology, Dalian 116024, China.
Email: zhaoxf@dlut.edu.cn

With the advancement of cameras and other image capturing technologies, machine-vision-based damage detection methods have developed rapidly, simply because machine vision can be adapted to diverse and complex engineering environments with high precision and high speed. A common approach is to feed image data streams to image processing algorithms for extracting abnormal features indicative of damage or change relative to a healthy baseline. Currently, machine vision has been applied in many fields, such as bridge monitoring,[6–8] vehicle monitoring,[9,10] dam monitoring,[11] and structural cracking detection.[12–14] Overall, vision-based damage detection methods are intuitive and effective at monitoring structural cracks, corrosion, and other common types of damage.

In recent years, with the rapid developments of deep learning, various algorithms have emerged with the ability to classify and recognize structural features at high accuracy. This enables automation and eliminates the need to manually process and compare processed image results with an assumed constant baseline. In fact, to date, deep learning has been widely used in medical diagnosis,[15–18] data mining,[19] autopilot,[20–23] and literary translation,[24,25] which have demonstrated remarkable results and success. In particular, significant achievements have been made in image classification largely due to the remarkable performance of convolutional neural networks (CNNs). For example, in 2017, Cha and Choi[26] proposed a crack recognition method based on CNN and introduced deep learning to the field of SHM for the first time. Their results showed that this method can meet the engineering needs and has high recognition accuracy. Therefore, combining deep learning with machine vision can be readily applicable for damage identification in civil engineering structures.

One of the most important structural components in buildings is its connections, where bolt damage can adversely affect structural safety and performance. Currently, many researchers have used machine vision to study the problem of damage in the form of bolt loosening. For example, Hough transform was used to identify the bolt edge, and the degree of bolt damage was judged by identifying the rotation angle of the bolt edge.[27] The length of the bolt was also assessed by support vector machine to determine the degree of bolt looseness.[28,29] While all of these methods have high accuracy, they all need to artificially extract damage characteristics through many intermediary steps. In addition, this procedure suffers from greater subjectivity when extracting damage characteristics. Thus, it is very difficult to actualize automated end-to-end damage monitoring using these aforementioned techniques. Based on the present situation, this article proposes a new bolt looseness damage detection method using deep learning.

This article combines machine vision and deep learning to propose a new monitoring method that can realize automated, end-to-end, bolt looseness damage detection without having to rely on feature extraction. First, a test structure was designed, and image datasets were collected using a typical smartphone. The dataset was separated such that 64% of it was used for training, 16% for validation, and 20% for testing. Then, a faster region–based convolutional neural network (Faster R-CNN) algorithm was trained and evaluated using the aforementioned image datasets. Second, this study then examined its applicability for damage detection by considering the extension of the bolt as a potential damage state of interest. In order to verify the accuracy of the new method, images of different bolts tightened to various heights and angles were identified. Third, to further introduce realism, the effect of lighting conditions on recognition accuracy of the detection method was analyzed. It was found that images that were taken in poorly lit conditions made damage detection more challenging; nevertheless, this method still showed strong detection capabilities under different lighting conditions. Then, to validate this method, four images of loosened bolts that were acquired from other structures were successfully detected by the trained model. Finally, in order to meet the needs of real-time monitoring, the trained model was connected with a webcam to detect and locate bolt looseness damage status. In summary, the new method not only enabled high precision and accurate damage detection, but it also met the requirement of real-time monitoring, which is a necessary step for realizing automated, end-to-end, damage monitoring.

Experimental details

Experimental test structure

This article proposes a new detection method that combines deep learning with machine vision to identify and locate bolt looseness damage. In this method, looseness of the bolt was defined by the extension length of the screw (or threaded bolt). Two states were defined: tight (intact) and loose (damaged). Figure 1 shows the test structure employed, where multiple bolts of different states could be introduced simultaneously. The test structure consisted of three iron plates, in which two plates were used as the supports, and the other large, horizontal plate was used for installing nine M28 bolts: the horizontal plate with length (L) of 400 mm, width (W) of 240 mm, and thickness of 10 mm, and the side plates with thickness of 20 mm, width (W) of 240 mm,

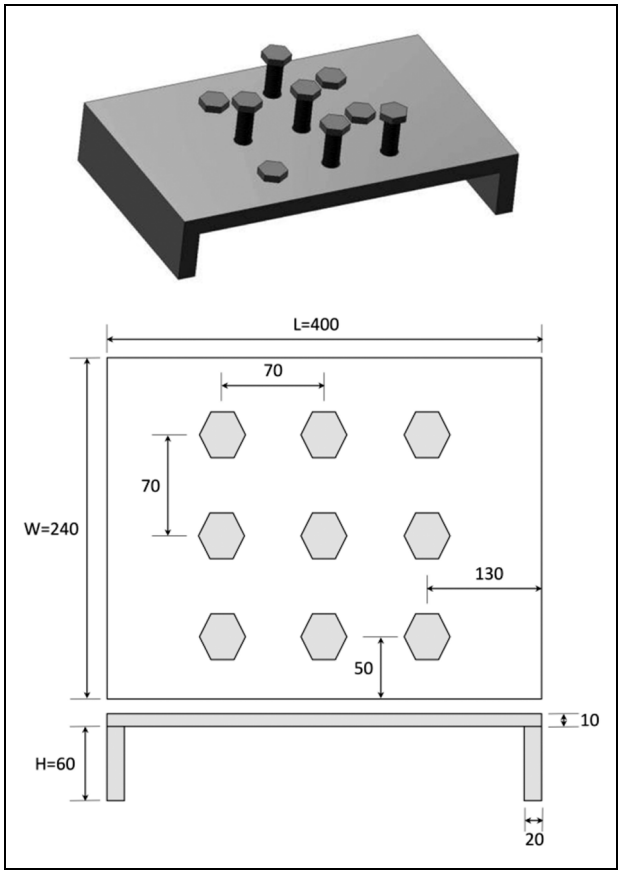

Figure 1. Schematic diagram of experiment structure.

and height (H) of 60 mm. In addition, the bolts distribution is shown in Figure 1. The spacing between bolts was 70 mm. Then, the Faster R-CNN algorithm was trained and assessed using image datasets obtained from this test setup. Thereafter, the trained model was used to detect the bolt looseness or damage.

Image datasets

The image datasets of the test structure that were used in this article were collected as shown in Figure 2. Initially, when a bolt is tightened, the length of the extend screw was 0 cm. Then, by loosening the bolt, the length of the extended portion of the screw was $0 \sim 3$ cm. Smartphones were used to acquire images of the bolts and test structure, and these images were taken at various angles and distances. The specifications of the onboard camera of smartphone are shown in Table 1. A total of 300 photos were acquired (4032×3016 pixels). In order to increase the training speed of the deep learning algorithm, all of the pictures were uniformly converted to 640×478 pixels by the tool (Format Factory), and the image file format is jpg. The datasets were then separated into 64% for training (192 images), 16% for validation (48 images), and 20% for testing (60 images). In addition, a tagging tool (LabelImg) was used to tag all images as shown in Figure 2, and the webpage of the tagging tool is https://github.com/tzutalin/labelImg.

Hardware and software configurations

The training of deep learning networks requires ample computing power. Thus, a Dell 7810 workstation was used for training the model. The workstation featured NVIDIA GeForce GTX 1080 Ti (with 11 GB of memory) and ran software such as the Ubuntu system, CUDA8.0, cuDNN6.0, python3.6, and tensorflow1.4, among others. The deep learning network used in this study is Faster R-CNN, as mentioned previously, and the CNN framework is VGG16.

Methodology

The central hypothesis of this study is that CNNs can be employed to extract structural damage features, such as bolt loosening, from image datasets. However, the problem of image positioning needs to be solved first. In the past, sliding windows were mostly used to divide an intact picture into multiple smaller pictures. Then these small pictures were classified and recognized for image positioning, which can be inefficient. Instead, locating objects can be achieved by integrating an object detection algorithm with CNN. There are two main approaches to implement this. The first approach is to treat it as a regression problem, and the second approach is to treat it as a region proposal problem. The Faster R-CNN used in this study belongs to the second approach. Faster R-CNN can not only accurately find the location of an object, but it can also identify the category of the object.[30–32]

CNNs

Convolutional layer. Convolution layer is one of the most important components in CNN. Each convolution layer contains multiple convolution kernels, which can extract features from the input image. The two-dimensional convolution is shown in Figure 3. Suppose that the input image is a matrix of dimensions 5×5, the convolution kernel is a matrix of dimensions 3×3. The first convolution operation ($\otimes$) begins with the upper left corner of the image. The parameters of the convolution kernel are multiplied by the pixel value of the coverage area. Then, the multiplied values are summed. When the bias term of the layer is not 0, the bias should be added to the summed values. As shown in the figure, the input image is convoluted by the convolution kernel from left to right and from top to

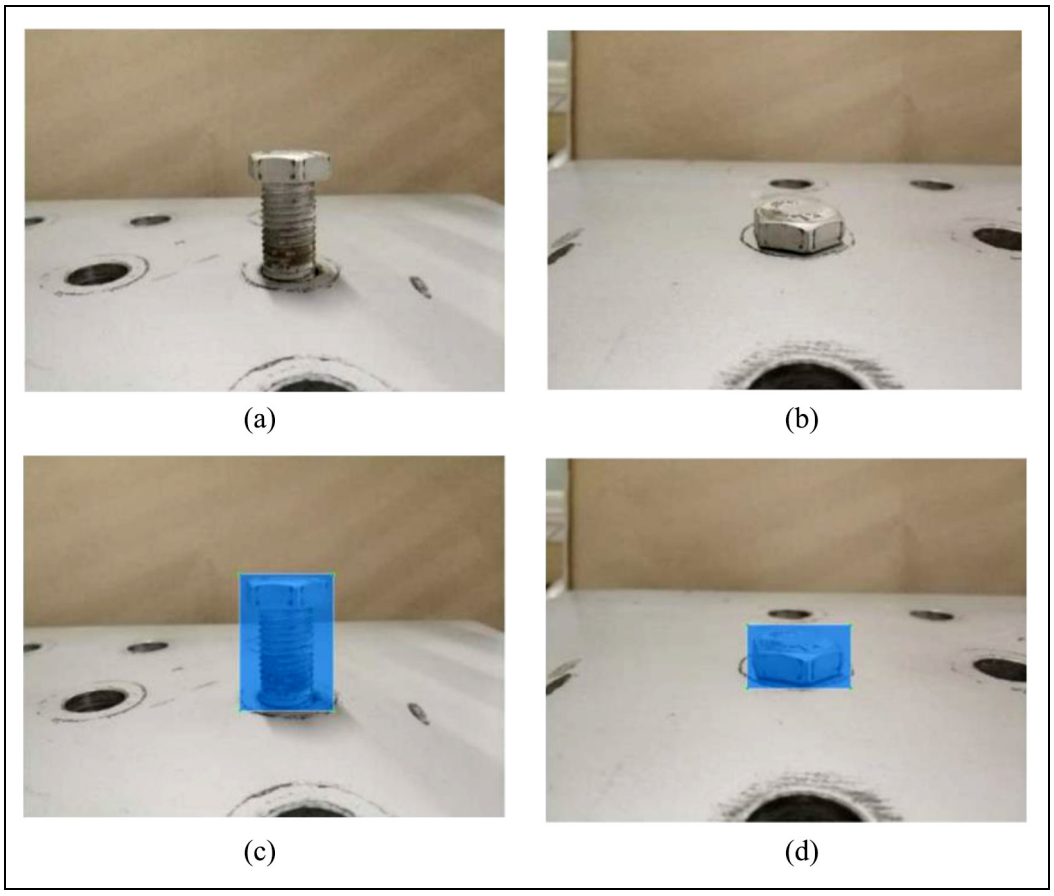

Figure 2. (a) Loosened and (b) tightened bolts, as well as the corresponding tagged images for the (c) damaged and (d) intact bolts, are shown.

Table 1. Smartphone camera specifications.

Parameters	Value
Size	4032 × 3016 pixels
Vertical resolution	72 dpi
Horizontal resolution	72 dpi
Bit depth	24
Aperture	f/1.8

bottom according to the stride. When the stride is 1, the convolution kernel moves one pixel to the right or down every time. Finally, the convolution feature of dimensions 3×3 is outputted, and the result is taken as the input of the next layer. The parameters of convolution kernel are learned through network training, and the initial parameters of convolution kernel are generated randomly.

Activation layer. Activation layer is also called nonlinear mapping layer. Activation function can increase the expressive ability of the whole network and form the complex functions. In traditional neural networks, sigmoid function is often used as the activation function. However, the sigmoid function can easily cause the saturating gradient problem. To avoid the saturating gradient problem, the rectified linear unit (ReLU) was introduced into the CNN.

Pooling layer. The pooling layer is also referred to as a down-sampling layer, and the size of output after pooling operation is smaller than the size of input. The pooling operations contain mean pooling and max pooling. The pooling layer is different from the convolution layer, and it has no parameters to learn. In the process of pooling operations, you only need to specify the type of pooling and the size of kernel and stride. The two-dimensional pooling operation is shown in Figure 4.

Fully connected layer. Fully connected (FC) layer is used as the classifier in CNN, and the learned features can be mapped to the sample markup space through this layer. It is noteworthy that the FC layer is also implemented by convolution operation.

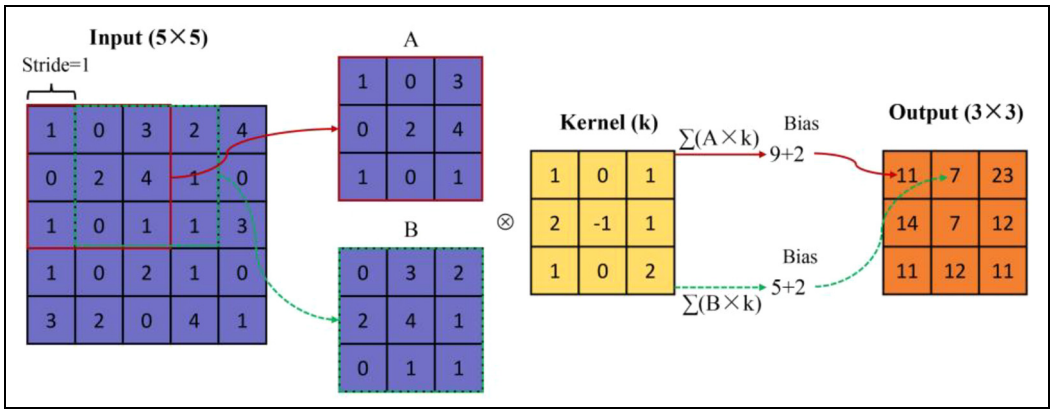

Figure 3. Convolution example.

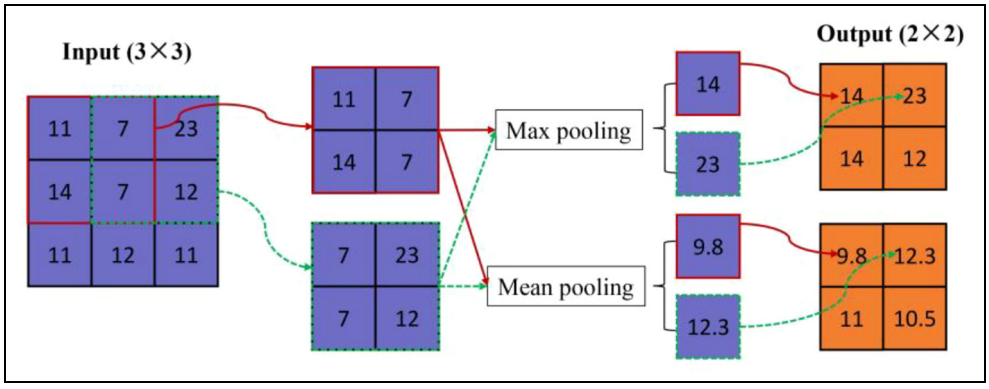

Figure 4. Pooling example.

Objective function. The objective function is also called loss function or cost function. It is used to measure the error between the predicted value and the sample value. In general, the softmax function is treated as the loss function for the classification problem.

R-CNN

In the field of object detection, R-CNN[30] is a very important algorithm. The architecture of R-CNN can be divided into four sections: candidate region selection, feature extraction, classification, and regression. The overall framework of the R-CNN is shown in Figure 5.

Candidate region selection. In order to locate the position of the object, the image is usually divided into many small areas, and then these areas are inputted into the detection model. These small areas can be obtained by sliding window method. However, the amount of data generated by this method is huge. To solve this problem, some region proposal generation methods were proposed. Candidate region selection is implemented through the selective search method. First, the picture is divided into many small areas. Second, the adjacent areas are continuously aggregated by the principle of similarity. The similarity includes color similarity, texture similarity, size similarity, and overlapping similarity. Finally, 1000 ~ 2000 candidate regions can be generated. These regions are processed to a uniform size as the input of CNN.

Feature extraction. After these regions are inputted into a deep CNN, the feature extraction can be carried out. The deep CNN used in this study is VGG16-Net, which includes 13 convolutional layers, 13 ReLU layers, 5 max pooling layers, 3 FC layers, and 1 softmax layer. For all the convolutional layers, kernel size is 3 and pad is 1. For all pooling layers, kernel size is 2 and stride is 2. The size of the input image is 224 × 224 pixels. The kernel size of 3 increases the ability of nonlinear expression, thereby making the segmentation plane more separable. VGG16-Net uses min-batch gradient descent to optimize multinomial logistic regression. R-CNN was trained on a model that was already well trained, such as VGG-Net, ZF-Net, and Res-Net.

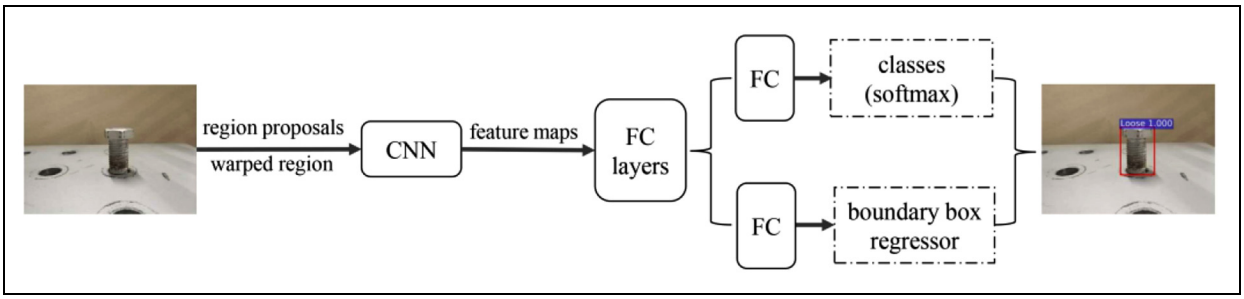

Figure 5. The R-CNN framework.

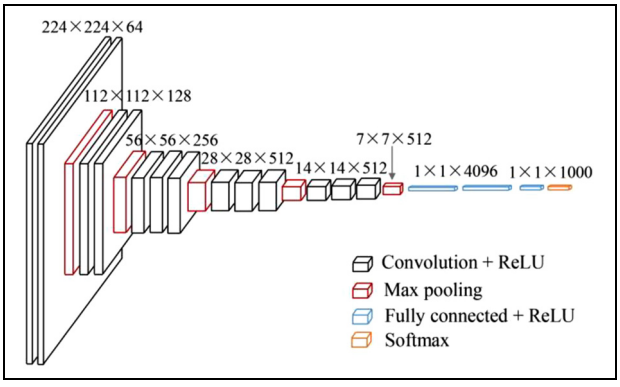

Figure 6. Microarchitecture of VGG16-Net.

The network model used in this article is VGG16-Net, as was mentioned earlier and shown in Figure 6.

Classification and regression. The feature map and proposals are inputted into the FC layer. One FC layer is the softmax, which can be used for classification; the other FC layer is the boundary box regression, which can be used for location. Finally, the object in the image can be identified and located. Although R-CNN has made great achievements in the field of object detection, it needs to select candidate regions before the feature extraction. And for traditional CNN, its input size is fixed, which will change the size and aspect ratio of regions in the standardization process, and every candidate region needs to enter CNN for calculation, which will lead to repeated calculation.

Faster R-CNN

Faster R-CNN[31] is presented based on the Fast R-CNN, and Faster R-CNN contains the Fast R-CNN and the region proposal networks (RPNs). The two networks share some parameters. The overall framework of the Faster R-CNN is shown in Figure 7.

RPN. The RPN is used to generate region proposals in Faster R-CNN, which greatly improves the generation speed of region proposals. The RPN is illustrated in Figure 8. To generate region proposals, a small network is slid over the convolution feature map output by the last shared convolution layer. Each point on the feature map will match those of the nine detection boxes, and this small network is fully connected to a spatial window of the input convolution (Conv) feature map.[32] Each sliding window is mapped to a lower

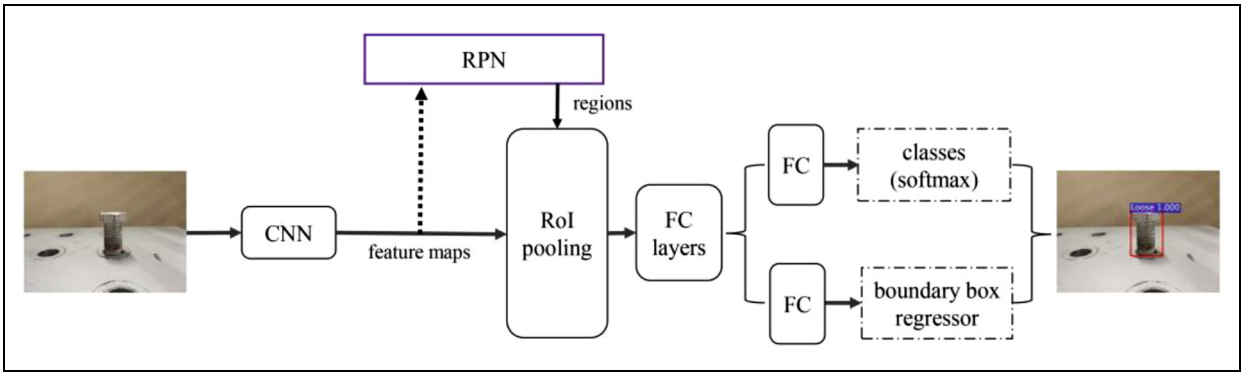

Figure 7. The Faster R-CNN framework.

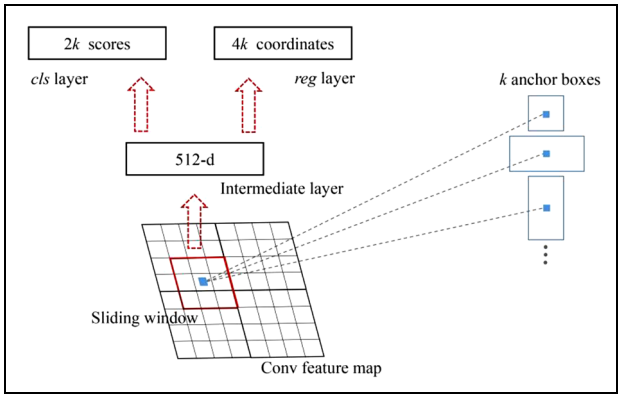

Figure 8. Region proposal network.

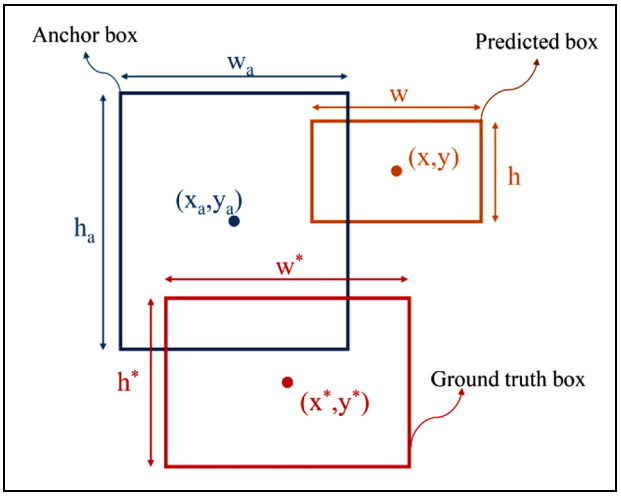

Figure 9. The geometry of a predicted box, an anchor box, and a ground-truth box.

dimensional vector (intermediate layer). After that, the vector is fed into two FC layers. One is regression layer (reg), and another is classification layer (cls). For each sliding window, k region proposals were proposed, so the regression layer has $4k$ outputs encoding the coordinates of k boxes. The classification layer outputs $2k$ scores that estimate probability of object/not-object for each proposal.[32] Then, the FC layer and the classification layer are used to calculate the category of each region. At the same time, bounding-box regression layer is used to calculate the offset of each object proposal position for obtaining a more accurate object detection proposal.

Loss function is used to measure the error between the predicted value and the sample value. When the RPN was trained, the loss function is used to measure the error between the predicted box and ground-truth box. The loss function is defined as

$$L(\{p_i\},\{t_i\}) = \frac{1}{N_{\text{cls}}}\sum_i L_{\text{cls}}(p_i,p_i^*) + \lambda\frac{1}{N_{\text{reg}}}\sum_i p_i^* L_{\text{reg}}(t_i,t_i^*) \tag{1}$$

The index of an anchor in a min-batch is i, p_i is the predicted probability of anchor i being an object, and p_i^* is the ground-truth label.[31] When p_i^* is 1, the anchor is positive, and when p_i^* is 0, the anchor is negative; t_i is a vector representing the four parameterized coordinates of the predicted bounding box; t_i^* is a vector representing the four parameterized coordinates of the ground-truth box; L_{cls} is the classification loss; and L_{reg} is the regression loss. When the anchor is a positive anchor, the regression loss is 0. The constants N_{cls} and N_{reg} are used as the normalization of classification loss and regression loss. λ is the balancing weight. The four parameterized coordinates can be calculated as

$$\begin{bmatrix} t_x, & t_y \\ t_w, & t_h \\ t_x^*, & t_y^* \\ t_w^*, & t_h^* \end{bmatrix} = \begin{bmatrix} (x-x_a)/w_a, & (y-y_a)/h_a \\ \log(w/w_a), & \log(h/h_a) \\ (x^*-x_a)/w_a, & (y^*-y_a)/h_a \\ \log(w^*/w_a), & \log(h^*/h_a) \end{bmatrix} \tag{2}$$

where (x, y) is the center coordinates of the predicted box, (x_a, y_a) is the center coordinates of the anchor box, and (x^*, y^*) is the center coordinates of the ground-truth box. w is the width of the predicted box, w_a is the width of the anchor box, and w^* is the width of the ground-truth box. h is the height of the predicted box, h_a is the height of the anchor box, and h^* is the height of the ground-truth box. The predicted box, anchor box, and ground-truth box are shown in Figure 9. This can be seen as bounding-box regression from an anchor box to a nearby ground-truth box. The detailed regression process of the bounding box can be found in the research of Girshick et al.[30]

Fast R-CNN. In Faster R-CNN, the RPN is used to generate region proposal and the Fast R-CNN[32] is used to locate and classify object. The convolutional features are shared between the RPN and Fast R-CNN. The overall framework of the Fast R-CNN is shown in Figure 10. A region of interest (RoI) pooling layer can extract a fixed-size feature vector from the feature map. Then, the fixed-size feature vector is inputted into FC layers to calculate the positions of predicted box and classify the objects in the boxes.[33]

As can be seen from Figure 8, a Fast R-CNN has two output layers (softmax layer and regression layer).

The softmax layer outputs the probability, p, over $K+1$ classes. The regression layer outputs the

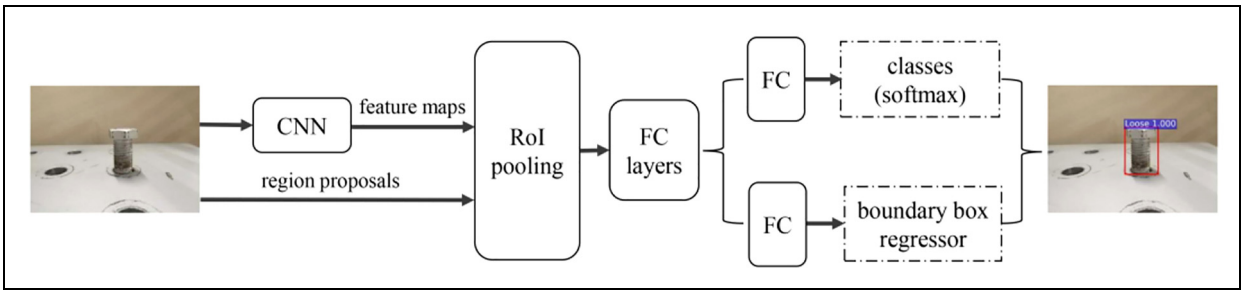

Figure 10. The Fast R-CNN framework.

information of the bounding box. When class is k, (t_x^k, t_y^k) is the center coordinates of the bounding box, t_w^k is the width of the bounding box, and t_h^k is the height of the bounding box. Each training RoI is labeled with a ground-truth class u and a ground-truth bounding-box regression target v. We use a multi-task loss L on each labeled RoI to jointly train for classification and bounding-box regression[31]

$$L(p,u,t^u,v) = L_{\text{cls}}(p,u) + \lambda[u \geqslant 1]L_{\text{loc}}(t^u,v) \quad (3)$$

in which $L_{\text{cls}}(p,u) = -\log p_u$ is log loss for true class u, λ is the hyper-parameter, and L_{loc} is the loss for bounding-box regression

$$L_{\text{loc}}(t^u,v) = \sum_{i\in\{x,y,w,h\}} \text{smooth}_{L_1}\left(t_i^u - v_i\right) \quad (4)$$

$$\text{smooth}_{L_1}(x) = \begin{cases} 0.5x^2 & if\,|x|<1 \\ |x| - 0.5 & \text{otherwise} \end{cases} \quad (5)$$

In order to sharing convolutional features between the two networks, a pragmatic four-step training algorithm was developed to learn shared features via alternating optimization. First, the RPN network is first trained to get a series of region proposals. Second, Fast R-CNN began to be trained. Third, the RPN network is trained again. Fourth, Fast R-CNN is trained for the second time. This training process repeated a total of two times. The reason of only repeating two times was that the training effect was not greatly improved as the number of repetitions was increased further.

Bolt damage object detection based on deep learning

Training process

In this study, Faster R-CNN based on TensorFlow frame was used to identify and locate bolt looseness damage. The aforementioned parameters were shared with the RPNs and object detection algorithm. Some training parameters are shown in Table 2. Intersection over union (IOU) is an evaluation metric used to measure the accuracy of an object detector on a particular dataset. IOU is simply a ratio of the area of overlap and the area of union between the predicted bounding box and the ground-truth bounding box. In order to ensure convergence of the training results, three iterations were attempted to train the datasets. The training results are shown in Table 2, and total loss is plotted with respect to the total number of iterations as shown in Figure 11.

Table 2. Training parameters.

Parameters	Value
Learning rate	0.001
Momentum	0.9
Weight decay	0.0005
The anchor scales for RPN	8, 16, 32
The anchor ratios for RPN	0.5, 1, 2
IOU	0.3

RPN: region proposal network; IOU: intersection over union.

The average precision (AP) results are summarized in Table 3, where the mean average precision (mAP) was found to be 0.9503, which is acceptable for engineering applications. The training process of the model is essentially the process of finding the optimal value by the gradient descent method. The training accuracy increases with the increase of iteration times and finally achieves a smooth state. At this point, the obtained value is close to the optimal value. However, it may be affected by the learning rate, and the obtained value oscillates around the optimal value. Thus, the training accuracy is also fluctuating in a small range. The detection accuracy of 5000 times is 0.9503, and the detection accuracy of 10,000 times is 0.9260. The accuracy of 10,000 times is lower than 5000 times, but the difference is very small. What's more, the detection accuracy of the model must meet the needs of the engineering. The

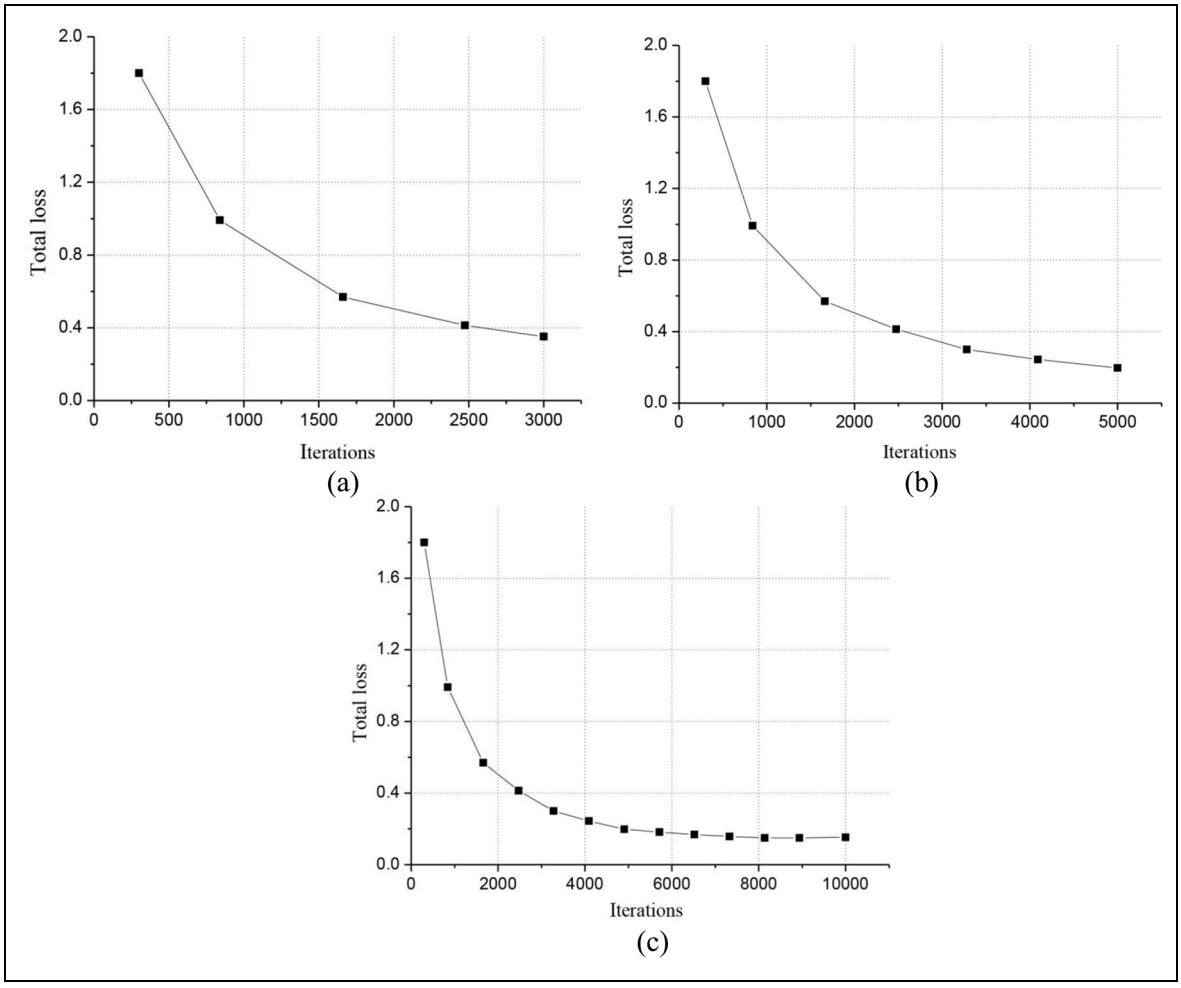

Figure 11. Total loss under different iterations: (a) 3000 iterations, (b) 5000 iterations, and (c) 10,000 iterations.

Table 3. AP corresponding to different iterations.

Iterations	AP for tight	AP for loose	mAP	Training time
3000	0.9100	0.9091	0.9095	674.8 s
5000	0.9005	1.0000	0.9503	1118.5 s
10,000	0.8559	0.9960	0.9260	2239.8 s

AP: average precision; mAP: mean average precision.

detection accuracy of 5000 times is 0.9503. It can meet the needs of the engineering. Based on these results, it was decided that 5000 iterations were used for the model.

Bolt looseness damage detection under different conditions

Minimum resolution for bolt looseness detection. It was mentioned earlier that loosened bolts were defined as those with a screw height of 3 cm. However, the model which was trained by deep learning has the potential to generalize the definition of loosening, which can therefore be used to identify loosened bolts with different screw heights. Thus, in this section, a series of images with different screw heights were tested for finding the minimum resolution for bolt looseness detection.

Representative test results are shown in Figure 12. The four images with screw heights of 2, 1, 0.5, and 0.4 cm were identified in Figure 12. Among them, the loosened bolts with screw heights of 2, 1, and 0.5 cm

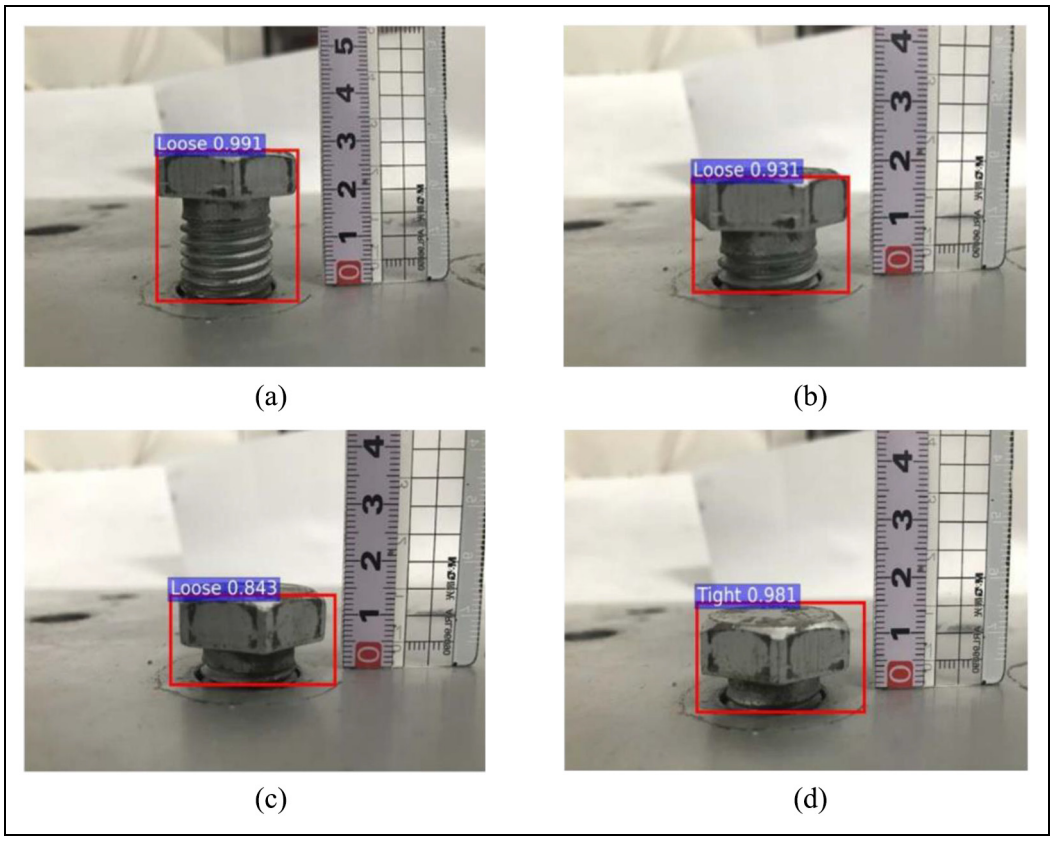

Figure 12. The detection result of different screw heights: (a) 2 cm, (b) 1 cm, (c) 0.5 cm, and (d) 0.4 cm.

were correctly identified as "Loose," and it was found that their recognition accuracy decreased as the height decreases (which makes sense as these became more difficult to recognize visually). Therefore, the screw heights of bolts can affect bolt looseness recognition. On the other hand, the loosened bolt with a screw height of just 0.4 cm was mistakenly identified as "Tight." Based on test results obtained in this study, it can be concluded that the minimum resolution for bolt looseness detection was 0.5 cm, based on the parameters of testing. Nevertheless, even though the detection method was trained with screw heights of 3 cm, the technique was successfully generalized to detect loosened bolts of different screw heights up to a minimum of 0.5 cm. These results also indicated that the recognition effect of this method was not limited to just the characteristics of the datasets and possessed a strong learning ability.

Looseness detection under different angles. In order to verify that the training model could identify bolt looseness damage based on images acquired at different angles, bolts of different loosened heights (i.e. 3, 2, and 1 cm) were photographed at a low angle (0°), medium angle (0°–45°), and high angle (45°–90°), as shown in Figure 13. Then, these images were used to see if the Faster R-CNN framework could identify that these bolts were in fact loose.

The test results are shown in Figure 13. The images acquired at low and medium angles successfully identified that all three bolts were loose. On the contrary, for the high angle image, the bolt with a loosened height of 1 cm was incorrectly identified as being tight. These results are important, since it clearly shows that detection methods based on machine vision are often restricted by camera shooting angle, which can lead to some errors during damage detection and is unavoidable.[26] Overall, the precision of bolt damage detection based on this method is acceptable.

Looseness detection under different lighting conditions. In order to detect the stability of the training model, images taken under different lighting conditions were evaluated. These pictures are divided into two categories, namely, normal lighting and dark lighting. Using the image datasets, the individual bolts were identified, and the training results showed that the framework could locate and identify looseness damage

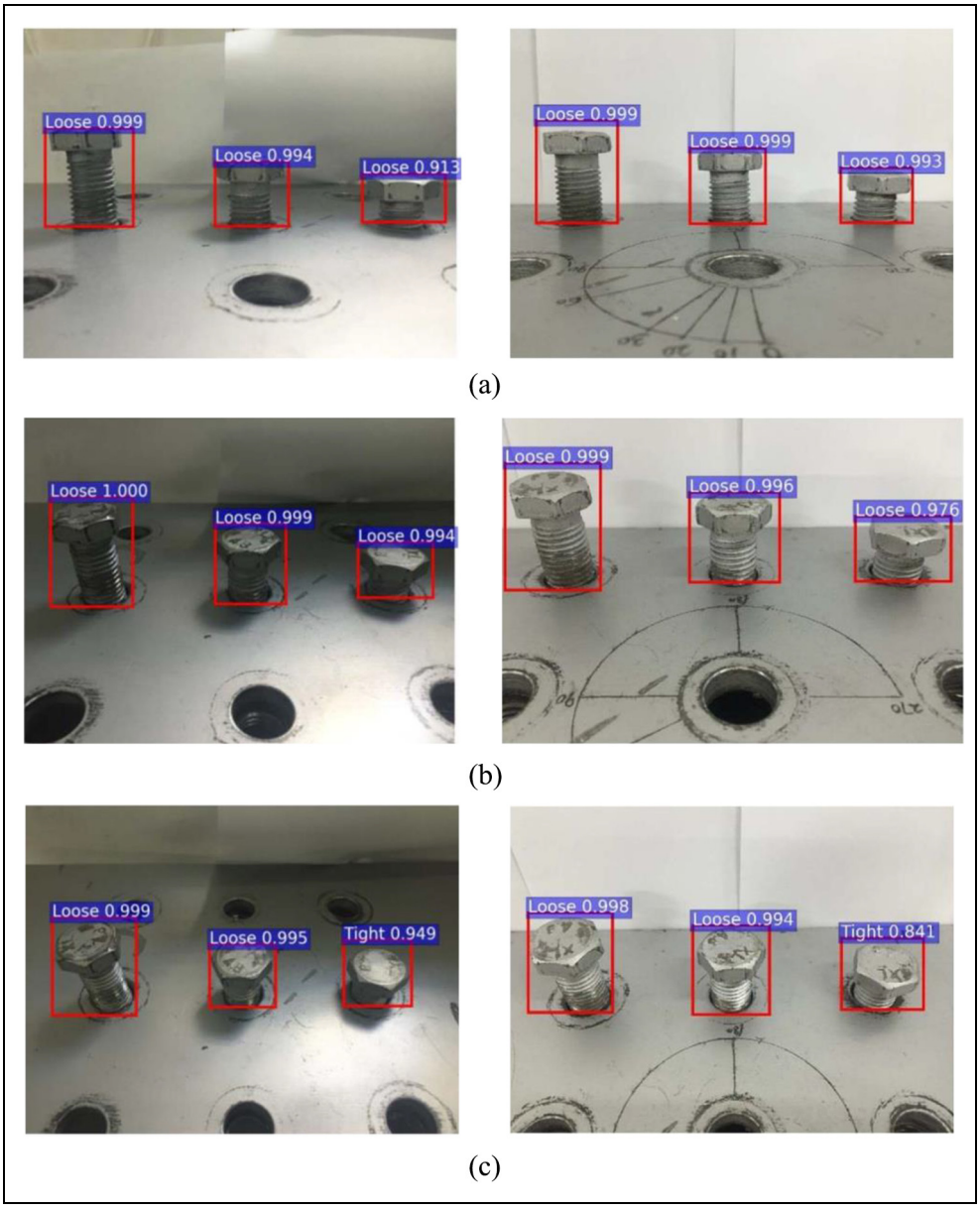

Figure 13. The detection result of different angles: (a) low angle, (b) medium angle, and (c) high angle.

of multiple bolts simultaneously. The looseness damage of nine bolts on the test structure was obtained from one image, where the image size was 3264×2448 pixel. Each image was tested separately, and the average time required for each image was 0.280 s. Some representative recognition results are shown in Figure 14. In addition, the statistics of the test results are summarized in Table 4.

As can be seen from Figure 14, the recognition accuracy of this method was still high under normal and dark lighting conditions. Table 4 computes statistical parameters that quantify recognition performance. In Table 4, P is positive (loose bolt), N is negative (tight bolt), TP is true positive, TN is true negative, FP is false positive, and FN is false negative. The identification accuracy, prediction accuracy, and recall rate for all kinds of cases encountered are listed in Table 4. The recall rate of all eight images was 100%; the precision of six images are 100%, while the other image had a damage detection error of one bolt, mainly because of the effect of shooting angle. In addition, as a result of the dark lighting condition, two bolts in the back row were not clearly photographed, so they were not identified. This caused the recognition accuracy to be 77.78% in Figure 14(c) and (d). Therefore, based on these results, poor lighting had a degree of influence on

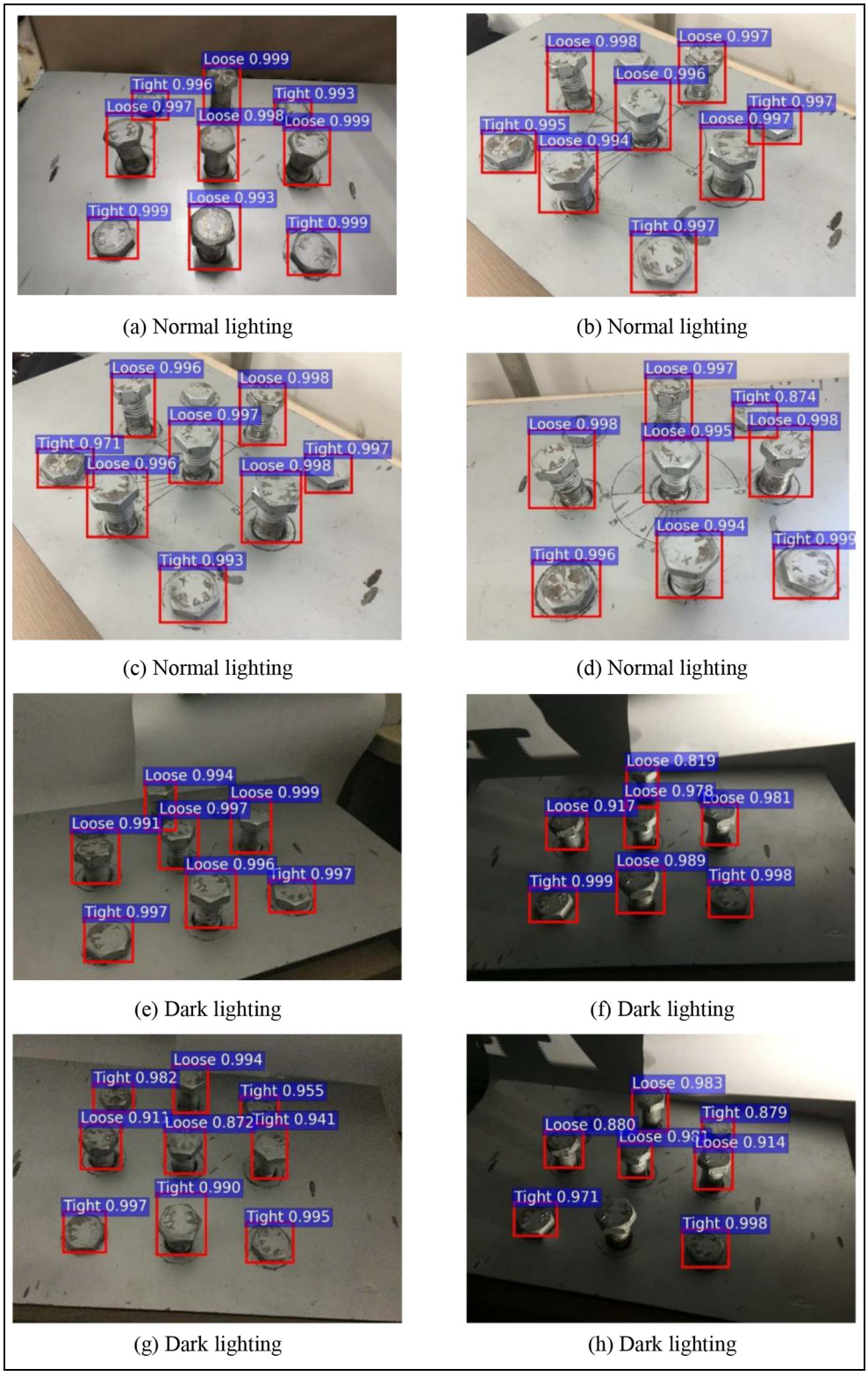

Figure 14. The detection results of bolt looseness under different lighting conditions: (a) to (d) normal lighting and (e) to (h) dark lighting.

Table 4. The detailed analysis of damage detection results based on different lighting conditions.

Test sample	P	N	TP	TN	FP	FN	Neglected	Accuracy	P		N	
									Precision	Recall	Precision	Recall
A	5	4	5	4	0	0	0	100%	100%	100%	100%	100%
B	5	4	5	3	0	0	1	88.89%	100%	100%	100%	100%
C	5	4	5	3	0	0	1	88.89%	100%	100%	100%	100%
D	5	4	5	3	0	0	1	88.89%	100%	100%	100%	100%
E	5	4	5	2	0	0	2	77.78%	100%	100%	100%	100%
F	5	4	5	2	0	0	2	77.78%	100%	100%	100%	100%
G	5	4	3	4	2	0	0	77.78%	60%	100%	100%	67%
H	5	4	4	3	0	0	2	77.78%	100%	100%	100%	100%

P: positive (loose bolt); N: negative (tight bolt); TP: true positive; TN: true negative; FP: false positive; FN: false negative.

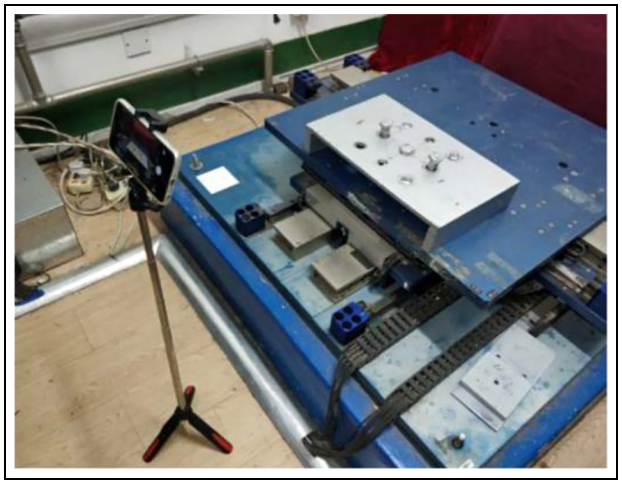

Figure 15. The shaking table test.

recognition precision of the framework, but it still satisfied the requirement of identifying and locating the bolts in the test structure. This experiment showed that the sharpness of the image had an important influence on recognition results, but shadow and poor lighting did not adversely affect recognition accuracy.

Looseness detection under different vibration conditions. In the real-world application, civil structures are always vibrating, which could cause photo blurry. In order to study the influence of blur, a shaking table test was completed. The image of test is shown in Figure 15. First, the experiment structure was placed on the shaking table. Second, the amplitude of the shaking table is set to 1 cm, and the frequency of the shaking table is set to 1 and 2 Hz. At the same time, a smartphone is used to capture images.

For the comprehensive testing, the three variables (shooting angle, vibration, and illumination) are studied together. The images were tested by the detection model. The detection results under the vibration frequency of 1 Hz are shown in Figure 16, and the detection results under the vibration frequency of 2 Hz are shown in Figure 17. Figure 16(a), (c), and (e) were taken under normal lighting; Figure 16(b), (d), and (f) were taken under dark lighting; Figure 16(a) and (b) were taken under low angle; Figure 16(c) and (d) were taken under medium angle; and Figure 16(e) and (f) were taken under high angle. The shooting condition of Figure 17 is the same as that of Figure 16. As can be seen from the detection results, blurred images have an impact on recognition results, and the degree of blurring in images increases as the vibration frequency increases. However, the detection model still has certain recognition ability for blurred images. Only loose bolt can be accurately identified in Figure 17(b), and the tight bolt has been ignored. It shows that large objects are more easily identified in blurred image, and the detection result of blurred image is greatly affected by illumination. In short, the effect of bolt looseness detection is still very good under low vibration frequency, but some bolts cannot be identified under high vibration frequency. This is also a limitation of this method.

Bolt looseness detection on other structures. In order to prove that the model could be generalized and used for different applications (even though training was based on bolts on the experimental test structure), images of loosened bolts were obtained from the two other completely different structures for validation. In order to better reflect the generalization ability of the model, a total of 10 images were collected. Because of space limitations, the detection results of four images have been shown in Figure 18, and the mean detection accuracies of 10 images are shown in Table 5. The mean–standard deviation accuracy can be calculated. It is only 4.207%. Because the bolts used for model training and those on the other structures are not of the same type, the

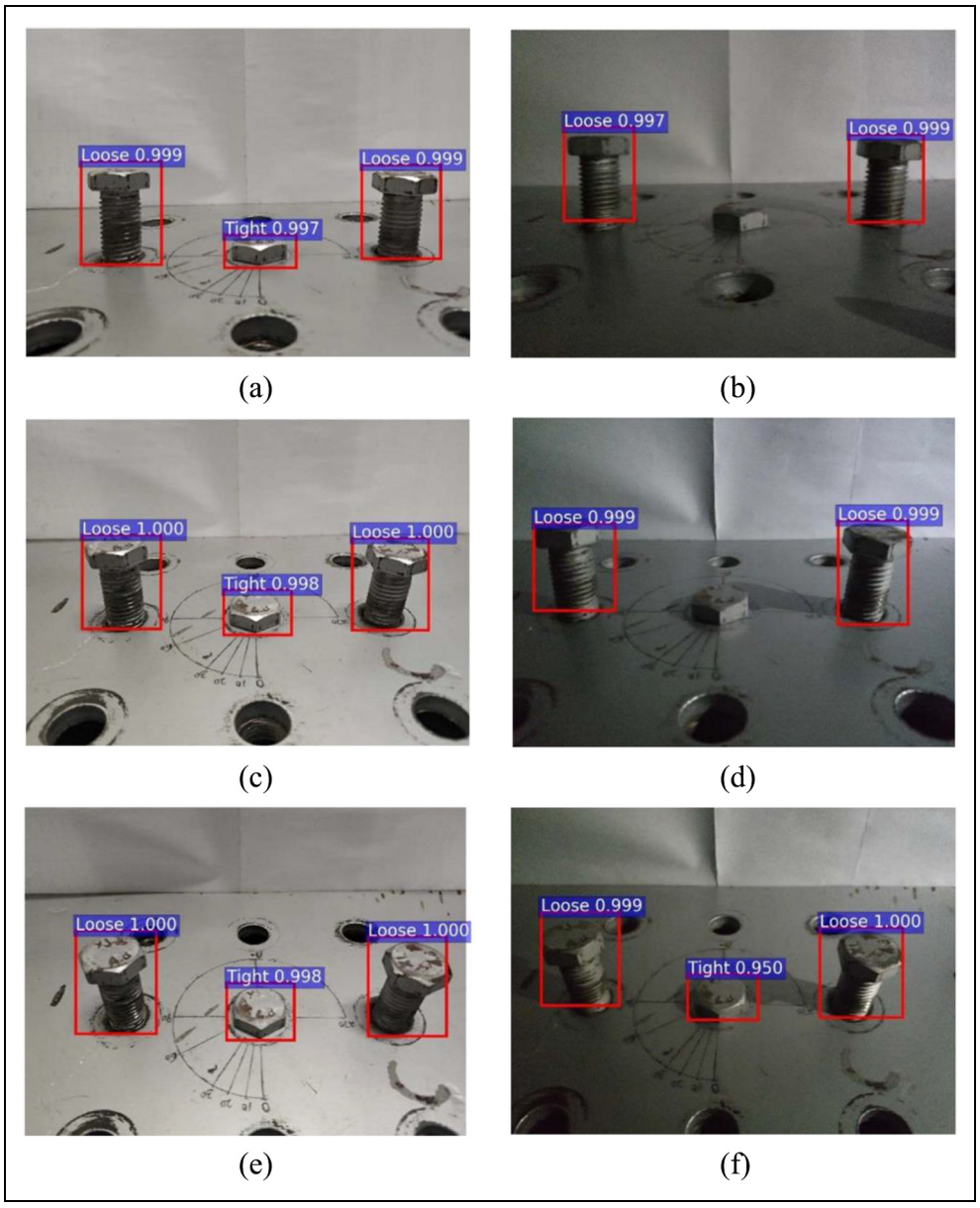

Figure 16. The detection results of bolt looseness under the vibration frequency of 1 Hz: (a), (c), (e) normal lighting; (b), (d), (f) dark lighting; (a), (b) low angle; (c), (d) medium angle and (e), (f) high angle.

localization accuracy of this method was inferior as compared to that of the previous section; nevertheless, high recognition accuracy was achieved. To further enhance bolt and damage detection capabilities, one could introduce different types of bolts to the training dataset in the future.

Real-time bolt looseness monitoring using a webcam

So far, bolt looseness damage detection was tested, verified, and validated by acquiring image datasets, training the model, and then subjected to testing using a different set of images. Obviously, this entire process was not continuous nor performed in real time. Therefore, the goal of this section was to demonstrate that real-time monitoring could be achieved by using a webcam to stream images, while the image datasets were processed in real time using the deep learning framework developed in this study. The training model was used to directly recognize the images collected by the webcam, and the experimental test setup is shown in Figure 19. This experiment employed an M1214-MP2 USB commercial webcam with a focal length of 12 mm,

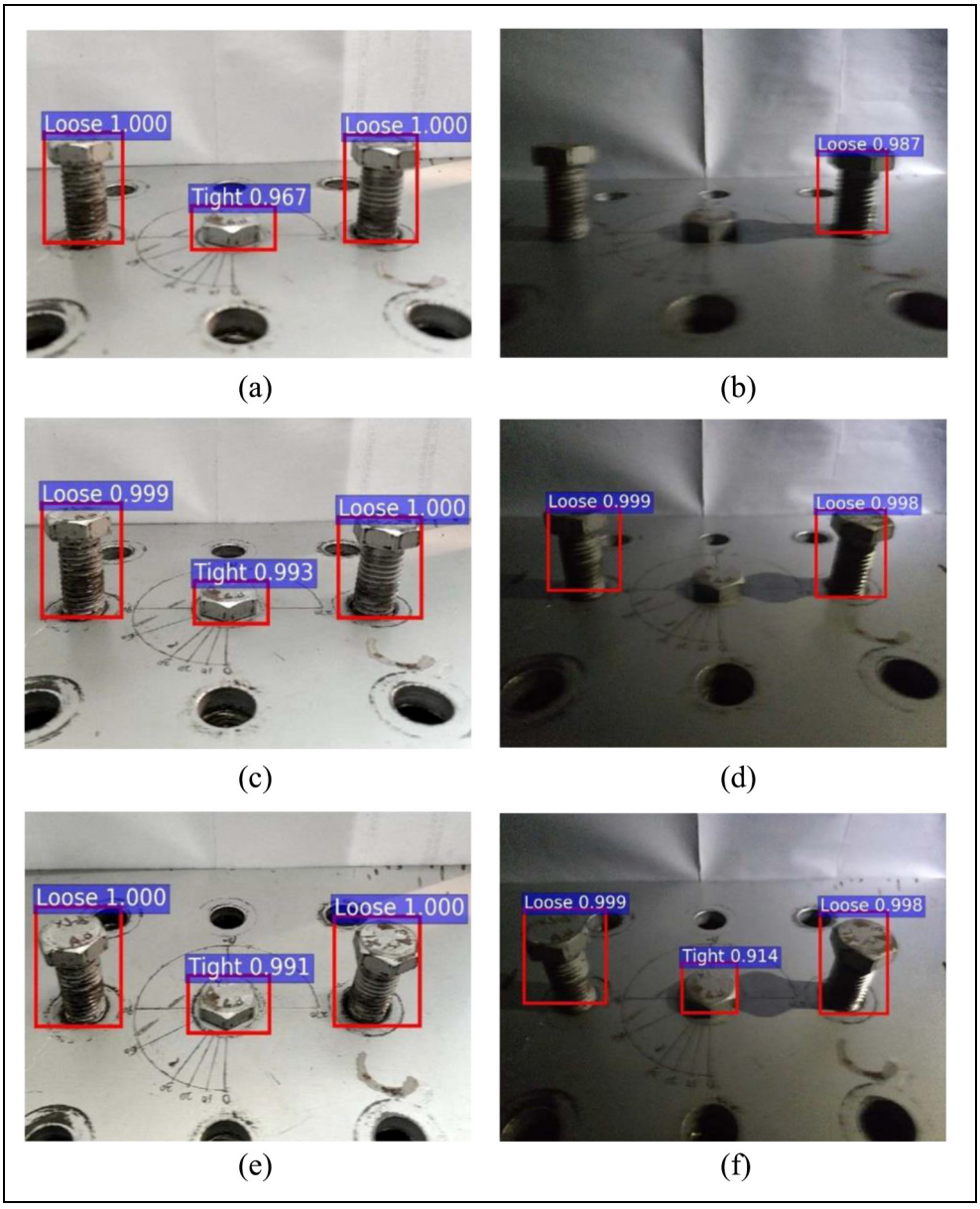

Figure 17. The detection results of bolt looseness under the vibration frequency of 2 Hz: (a), (c), (e) normal lighting; (b), (d), (f) dark lighting; (a), (b) low angle; (c), (d) medium angle and (e), (f) high angle.

maximum imaging size of 8.8 × 6.6, an aperture range of F1.4 ~ F16C, working distance of 0.15 ~ inf (m), and M30 0.5 × P0.5 filter thread.

It can be seen from Figure 19 that the state of the nine bolts was acquired in real time using the webcam, and the deep learning model could effectively locate and identify bolt looseness damage based on the streamed images. Six of the nine bolts on the experimental structure were in the loosened state, while the others were fully tightened. Overall, the bolt damage identification method based on deep learning could accurately identify and locate the nine bolts and their corresponding states. It was found that the identification probability of each bolt state was greater than 0.90. These results show that the detection method not only had high recognition accuracy, but it had the potential for real-time and online damage recognition, even when using a simple webcam. Therefore, one could begin to assemble a real-time bolt looseness damage monitoring system using the framework

Table 5. The detailed analysis of damage detection results based on other structures.

Test sample	Mean detection accuracy
1	0.949
2	0.903
3	0.822
4	0.924
5	0.912
6	0.969
7	0.966
8	0.904
9	0.919
10	0.897

investigated in this study. Such a system can provide early damage warning and enhance the safety of engineered structures.

Conclusion

This study proposed a new bolt looseness damage monitoring method based on deep learning. First, a test structure that accommodated nine bolts was fabricated, and bolts that were either tight or loosened (with an extended screw length of 3 cm) were randomly installed on the structure. Second, Faster R-CNN with an identification algorithm based on region proposals was

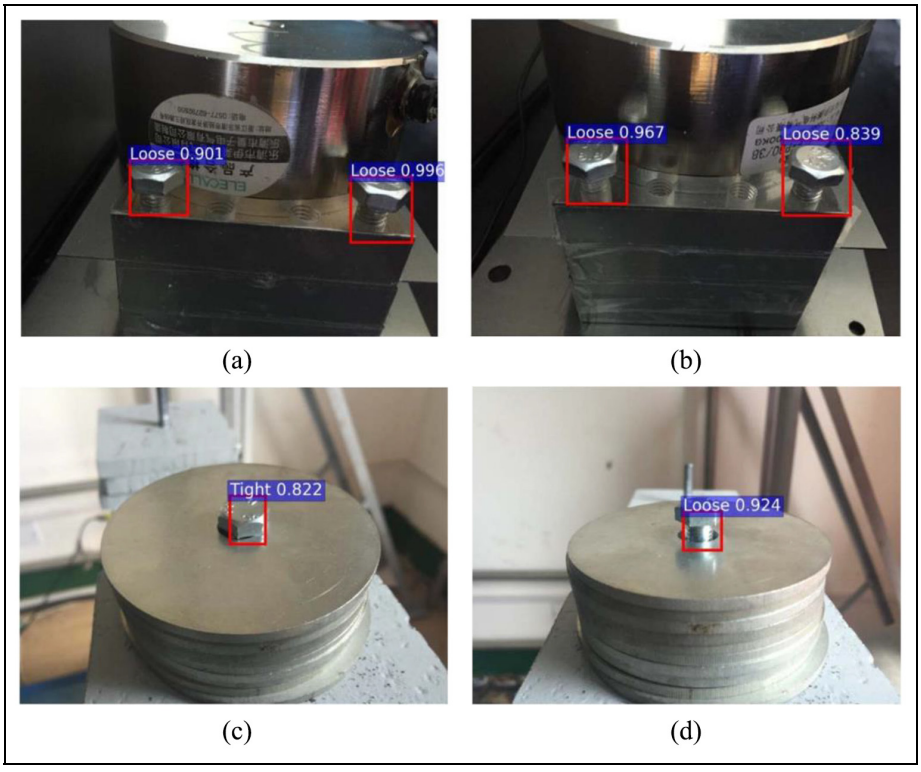

Figure 18. Bolt detection validation results when bolts were installed on different structures: (a), (b), (d) loose bolt and (c) tight bolt.

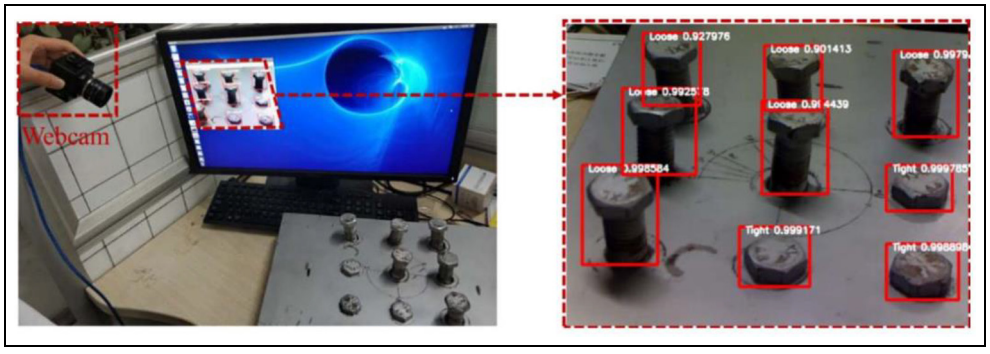

Figure 19. Bolt looseness real-time monitoring.

implemented for object detection and localization, which was then used to train image datasets of the test structure. Overall, it was found that the network did not require a large dataset to achieve good recognition effect. The test results showed that the training model had high recognition accuracy and an AP of 0.9503. Although the length of all the loosened bolts was 3 cm, the trained model could still accurately identify loosened bolts even if their lengths were just 0.5 cm. However, test results also showed that recognition accuracy could be compromised depending on the angle in which images were acquired. This is a common problem of machine vision and can be solved by rotating the camera to reduce the possibility of misclassification. Furthermore, images of loosened bolts acquired under different lighting conditions still produced excellent bolt recognition results, indicating that the training model had strong generalization ability and robustness. Finally, in order to validate real-time damage monitoring, a webcam was used to stream video images of the test structure with both loosened and tightened bolts. The deep learning framework developed in this study could identify and locate damage that appeared in the webcam's field-of-view in real time. In summary, the bolt looseness damage monitoring framework based on deep learning implemented and investigated in this study exhibited high recognition accuracy and could be used for real-time damage monitoring.

Acknowledgements

The work presented here was carried out in collaboration between all authors. X.Z. and Y.Z. contributed the conception of the bolt looseness detection method. Y.Z., X.S., K.J.L., W.S., and Z.X. carried out the bolt looseness detection based on deep learning. Y.Z. and K.J.L. analyzed the test data and prepared the manuscript.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

This research was supported by the Science and Technology Project of China Jiangsu Province Special Equipment Safety Supervision Inspection Institute in 2016 (KJ (Y) 2016008), National Natural Science Foundation of China (51479031), and the National Key R&D Program of China during the 13th Five-Year Plan Period (2016YFC0802002). K.J.L. was supported by the Jacobs School of Engineering, University of California–San Diego.

ORCID iDs

Kenneth J Loh https://orcid.org/0000-0003-1448-6251

Xuefeng Zhao https://orcid.org/0000-0002-1704-4021

References

1. Doebling SW, Farrar C, Prime MB, et al. Damage identification and health monitoring of structural and mechanical systems from changes in their vibration characteristics: a literature review. *Shock Vib Dig* 1996; 30(11): 2043–2049.
2. Housner GW, Bergman LA, Caughey TK, et al. Structural control: past, present, and future. *J Eng Mech* 1997; 123(9): 897–971.
3. Doebling SW, Farrar CR and Prime MB. A summary review of vibration-based damage identification methods. *Shock Vib Dig* 1998; 30(2): 91–105.
4. Zhou G-D and Yi T-H. Recent developments on wireless sensor networks technology for bridge health monitoring. *Math Probl Eng* 2013; 2013(3): 947867.
5. Federici F, Graziosi F, Faccio M, et al. An integrated approach to the design of wireless sensor networks for structural health monitoring. *Int J Distrib Sens N*. Epub ahead of print 22 March 2012. DOI: 10.1155/2012/594842.
6. Oh JK, Jang G, Oh S, et al. Bridge inspection robot system with machine vision. *Automat Constr* 2009; 18(7): 929–941.
7. Kim SW, Jeon BG, Kim NS, et al. Vision-based monitoring system for evaluating cable tensile forces on a cable-stayed bridge. *Struct Health Monit* 2013; 12(5–6): 440–456.
8. Luo L, Feng MQ and Wu ZY. Robust vision sensor for multi-point displacement monitoring of bridges in the field. *Eng Struct* 2018; 163: 255–266.
9. Trivedi MM, Gandhi T and McCall J. Looking-in and looking-out of a vehicle: computer-vision-based enhanced vehicle safety. *IEEE T Intell Transp* 2007; 8(1): 108–120.
10. Zhu Z, Xu G, Yang B, et al. VISATRAM: a real-time vision system for automatic traffic monitoring. *Image Vision Comput* 2000; 18(10): 781–794.
11. Zaczek-Peplinska J, Elżbieta Kowalska M, Malowany K, et al. Application of digital image correlation and geodetic displacement measuring methods to monitor water dam behavior under dynamic load. *J Civ Eng Archit* 2015; 9(12): 1496–1505.
12. Jahanshahi MR and Masri SF. A novel vision-based crack quantification approach by incorporating depth perception for condition assessment of structures. In: *International conference on computing in civil engineering*, Clearwater Beach, FL, 17–20 June 2012, pp. 531–536. Reston, VA: ASCE.
13. Shan B, Zheng S and Ou J. A stereovision-based crack width detection approach for concrete surface assessment. *KSCE J Civ Eng* 2016; 20(2): 803–812.
14. Jahanshahi MR and Masri SF. Adaptive vision-based crack detection using 3D scene reconstruction for condition assessment of structures. *Automat Constr* 2012; 22: 567–576.
15. Li R, Zhang W, Suk HI, et al. Deep learning based imaging data completion for improved brain disease diagnosis. *Med Image Comput Comput Assist Interv* 2014; 17(Pt 3): 305–312.

16. Nie L, Wang M, Zhang L, et al. Disease inference from health-related questions via sparse deep learning. *IEEE T Knowl Data En* 2015; 27(8): 2107–2119.
17. Liu S, Liu S, Cai W, et al. Early diagnosis of Alzheimer's disease with deep learning. In: *IEEE international symposium on biomedical imaging*, Beijing, China, 29 April–2 May 2014, pp. 1015–1018. New York: IEEE.
18. Hammerla NY, Fisher JM, Andras P, et al. PD disease state assessment in naturalistic environments using deep learning. In: *Twenty-ninth AAAI conference on artificial intelligence*, Austin, TX, 25–30 January 2015, pp. 1742–1748. Palo Alto, CA: AAAI Press.
19. Wlodarczak P, Soar J and Ally M. Multimedia data mining using deep learning. In: *Fifth international conference on digital information processing and communications*, Sierre, 7–9 October 2015, pp. 190–196. New York: IEEE.
20. Liu M, Niu J and Wang X. An autopilot system based on ROS distributed architecture and deep learning. In: *IEEE international conference on industrial informatics*, Emden, 24–26 July 2017, pp. 1229–1234. New York: IEEE.
21. Bojarski M, Del Testa D, Dworakowski D, et al. End to end learning for self-driving cars, 2016, https://arxiv.org/abs/1604.07316
22. Shalev-Shwartz S, Ben-Zrihem N, Cohen A, et al. Long-term planning by short-term prediction, 2016, https://arxiv.org/abs/1602.01580
23. Santana E and Hotz G. Learning a driving simulator, 2016, https://arxiv.org/abs/1608.01230
24. Castelvecchi D. Deep learning boosts Google Translate tool. *Nature* 2016, https://www.nature.com/news/deep-learning-boosts-google-translate-tool-1.20696
25. Faisal M and Manzoor S. Deep learning for lip reading using audio-visual information for Urdu language, 2018, https://arxiv.org/abs/ 1802.05521
26. Cha YJ and Choi W. Vision-based concrete crack detection using a convolutional neural network. In: Caicedo J and Pakzad S (eds) *Dynamics of civil structures*, vol. 2. Cham: Springer, 2017, pp. 71–73.
27. Park JH, Huynh TC, Choi SH, et al. Vision-based technique for bolt-loosening detection in wind turbine tower. *Wind Struct* 2015; 21(6): 709–726.
28. Ramana L, Choi W and Cha YJ. Fully automated vision-based loosened bolt detection using the Viola–Jones algorithm. *Struct Health Monit* 2018; 18: 422–434.
29. Ramana L, Choi W and Cha YJ. Automated vision-based loosened bolt detection using the cascade detector. In: Wee Sit E, Walber C, Walter P, et al. (eds) *Sensors and instrumentation*, vol. 5. Cham: Springer, 2017, pp. 23–28.
30. Girshick R, Donahue J, Darrell T, et al. Rich feature hierarchies for accurate object detection and semantic segmentation. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, Columbus, OH, 23–28 June 2014.
31. Ren S, He K, Girshick R, et al. Faster R-CNN: towards real-time object detection with region proposal networks. In: Cortes C, Lawrence ND, Lee DD, et al. (eds) *Advances in neural information processing systems*, vol. 28. Red Hook, NY: Curran Associates, Inc., 2015, pp. 91–95.
32. Girshick R. Fast R-CNN, 2015, https://arxiv.org/abs/1504.08083
33. Cha YJ, Choi W, Suh G, et al. Autonomous structural visual inspection using region-based deep learning for detecting multiple damage types. *Comput Aided Civ Inf* 2018; 33(9): 731–747.