Smart Structures and Systems, *Vol. 29, No. 1 (2022) 207-220*
https://doi.org/10.12989/sss.2022.29.1.207

Impacts of label quality on performance of steel fatigue crack recognition using deep learning-based image segmentation

Shun-Hsiang Hsu [1a], Ting-Wei Chang [2b] and Chia-Ming Chang*[2]

[1] *NCREE-NTUCE Joint Artificial Intelligence Research Center, No. 200, Sec. 3, Xinhai Rd., Da'an Dist., Taipei City 106219, Taiwan (R.O.C.)*
[2] *Department of Civil Engineering, National Taiwan University, No. 1, Sec. 4, Roosevelt Rd., Da'an Dist., Taipei City 106319, Taiwan (R.O.C.)*

(*Received April 30, 2021, Revised July 2, 2021, Accepted July 9, 2021*)

Abstract. Structural health monitoring (SHM) plays a vital role in the maintenance and operation of constructions. In recent years, autonomous inspection has received considerable attention because conventional monitoring methods are inefficient and expensive to some extent. To develop autonomous inspection, a potential approach of crack identification is needed to locate defects. Therefore, this study exploits two deep learning-based segmentation models, DeepLabv3+ and Mask R-CNN, for crack segmentation because these two segmentation models can outperform other similar models on public datasets. Additionally, impacts of label quality on model performance are explored to obtain an empirical guideline on the preparation of image datasets. The influence of image cropping and label refining are also investigated, and different strategies are applied to the dataset, resulting in six alternated datasets. By conducting experiments with these datasets, the highest mean Intersection-over-Union (mIoU), 75%, is achieved by Mask R-CNN. The rise in the percentage of annotations by image cropping improves model performance while the label refining has opposite effects on the two models. As the label refining results in fewer error annotations of cracks, this modification enhances the performance of DeepLabv3+. Instead, the performance of Mask R-CNN decreases because fragmented annotations may mistake an instance as multiple instances. To sum up, both DeepLabv3+ and Mask R-CNN are capable of crack identification, and an empirical guideline on the data preparation is presented to strengthen identification successfulness via image cropping and label refining.

Keywords: crack recognition; deep learning; image segmentation; label quality

1. Introduction

Structural health monitoring (SHM) can be defined as a process to identify damage for civil infrastructure through periodically spaced measurements and to further determine the state of health (Farrar and Worden 2007). The autonomous inspection for the measurements has been studied extensively in recent years because conventional methods are widely considered labor-intensive and time-consuming (Feng and Feng 2018). Considering that the number of aging structures increases and even parts of these structures may be close to the end of their lifespans, the autonomous inspection should be increasingly important in saving considerable cost (Song *et al*. 2017). Among all types of damage, cracks including fatigue cracks are crucial to structural safety. In the case of bridges, fatigue cracks mainly result from cyclic loading and are usually found on welding joints. These cracks would reduce the effective area of cross-section and can further result in brittle fracture which occurs without an omen (Albrecht and Wright 2000). Consequently, continuous inspections of fatigue cracks are paramount to provide forewarning of the fracture. In practice, the visual inspection is mostly adopted to locate defects and cracks, and the associated records from the inspection are photos. Taking advantage of the advancement in hardware for image acquisition, high-quality images can be easily obtained by lightweight devices. However, the difficulties of crack identification mainly hinder the automation in the visual inspection. In the past, various approaches combined with image processing were proposed to address this problem (Mohan and Poobal 2018). Overall, cracks would be filtered by hand-crafted features which may not be sufficiently discriminative. As a result, strategies for denoising such as shallow correction were implemented to prevent confusion with the background. Despite the long-term efforts, the applicability of these methods varies case by case, and the performance of crack identification seriously decreases in different scenarios. This outcome raises major concerns for practical use, and thus a more robust autonomous inspection method for crack identification is of need.

Deep learning-based methods such as convolutional neuron network (CNN) had outperformed other algorithms in various applications of computer vision including image segmentation and can be employed to detect surface cracks of structures. The methods of segmentation can be further classified into two groups: instance segmentation and semantic segmentation. The major difference between these two groups is that instance segmentation distinguishes each object, while semantic segmentation treats objects in the

*Corresponding author, Associate Professor,
E-mail: changcm@ntu.edu.tw
[a] Research Assistant, E-mail: b02501030@gmail.com
[b] Graduate Student

ISSN: 1738-1584 (Print), 1738-1991 (Online)

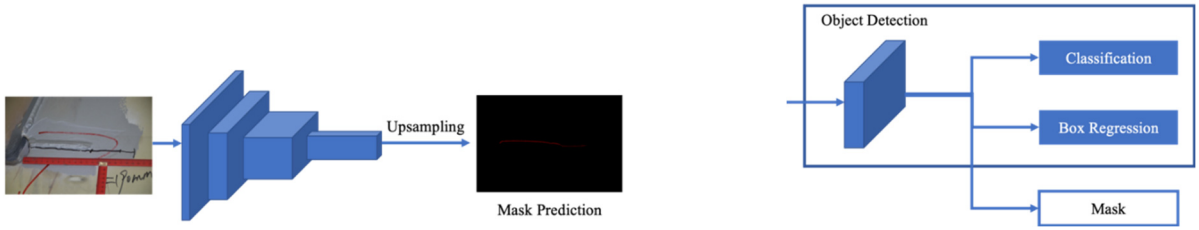
same class as a single label. For semantic segmentation, Long *et al*. (2015) applied fully convolutional network (FCN) to spatially dense prediction tasks (i.e., segmentation task) and achieved the state-of-the-art segmentation of public datasets. Dung (2019) implemented FCN to perform semantic segmentation on concrete crack images, and the model reached the average precision (AP) up to 90%. The utilization of fully convolutional computation had inspired future studies, and thus the improvement had been made by raising the significance of features or adopting a more efficient strategy for training. Then, Chen *et al*. (2017) proposed DeepLab, where the model combined atrous convolution (i.e., dilated convolution) and atrous spatial pyramid pooling (ASPP) to enlarge the receptive field, and the inference on public datasets is more accurate than that of FCN. Because of the efficiency and the performance, its advanced version, DeepLabv3, was presented to identify cracks on the surface of a tunnel in the paper (Song *et al*. 2019). By contrast, instance segmentation is more challenging than semantic segmentation because different instances need to be separated. He *et al*. (2017) presented Mask R-CNN, which was originated from Faster R-CNN (Ren *et al*. 2015) by adding a branch for mask prediction. This had built a typical framework that is detection-based for instance segmentation because of the stability of bounding box predictions and a minor increase in the number of parameters. Most important of all, the model can obtain state-of-the-art performance. Kalfarisi *et al*. (2020) had applied Mask R-CNN to crack segmentation, and the best Intersection-over-Union (IoU) is 37%. Therefore, DeepLabv3+ and Mask R-CNN are considered the most competitive algorithms and should be implementable to identify fatigue cracks in images.

As crack identification based on deep learning emerged, most studies focused on implementing various algorithms without detailed discussions on the quantity and quality of training data. However, training data also play an important role in model performance. For one thing, a limited amount of data may hinder the generalization of a model and result in potential overfitting. Despite the data augmentation, a dearth of the quantity can be insolvable unless more images are available. For another thing, data with poor labeling quality may cause arguable results and even yield a model that cannot converge. In particular, segmentation models require pixel-level annotations, and cracks usually occupy a tiny portion of images. As a result, manual mistakes are more frequent when labeling the data, and more labor hours are necessitated to avoid such incidents. Because model performance may largely depend on the quality of annotations, possible impacts should be explored to develop the most efficient strategy for labeling. In consequence, the pre-processing methods, image cropping and label refining, that can alter the quality of labels were applied to the dataset to examine their effects on the model performance. This paper concentrates efforts on not only maximizing model performance for crack identification but also clarifying the impacts of labeling quality on performance.

In this study, the experiments are conducted on the image dataset provided by the organizations of the International Project Competition for SHM (IPC-SHM 2020). The dataset is composed of 200 RGB images with fatigue cracks on welding joints, and only 120 of them are well-annotated. For comprehensive comparison, various combinations of training data and validation data are presented to conduct different experiments. Although another 160 unlabeled images were released after the competition, these images are not included for either training or validation, and only the results on these images are demonstrated for discussion. For the experiments, three strategies for image cropping and two different labeling policies are introduced to produce different label quality. The image cropping tests the effects of the crack pixels in an image while the label refining examines the crack detailing (i.e., rough marking or almost exact marking on cracks). As a result, six different datasets are employed to train two well-known segmentation models, DeepLabv3+ and Mask R-CNN. To fairly evaluate the image cropping and label refining effects, the hyperparameters used in these two models are the same, and the other settings are identical to those stated in the original papers (He *et al*. 2017, Chen *et al*. 2018). To sum up, DeepLabv3+ and Mask R-CNN trained on different datasets are evaluated to investigate the influence of labeling quality on performance. As found in the results, the best model can achieve mIoU as high as 75%.

2. Review of deep learning-based image segmentation

For SHM, structural safety is analyzed with the geometry of cracks (e.g., width, length) as well as its changes along with time. In practice, digital image processing was widely used to obtain the information. Additionally, pixel-wise identification is also a potential solution to addressing the problem, and the success of deep learning approaches in computer vision became more and more popular. Thus, research on deep learning-based image segmentation was reviewed. Afterward, state-of-the-art approaches were implemented to accurately position cracks on welding joints. To systematically review the related work, frameworks, where a model is built, had been surveyed, and only papers based on the mainstream framework were studied. For semantic segmentation, the most popular framework based on FCN was first introduced by Long *et al*. (2015), and the encoder-decoder structure in this framework comprises a series of downsampling and upsampling. As can be seen in Fig. 1(a), with the architecture, end-to-end learning can be realized because the model can accept an input of any size without additional processing and predict corresponding-sized outputs (i.e., classification maps). In that way, the number of output channels is equal to the number of categories, and the output value of each pixel is the probability for classification. Additionally, FCN is compatible with most modern CNN; therefore, the utilization of FCN becomes popular and is widely adopted to reach semantic segmentation. On the other hand, instance segmentation is simple and powerful to perform mask prediction on the result of object detection which is multiple bounding boxes



(a) Semantic segmentation (b) Instance segmentation

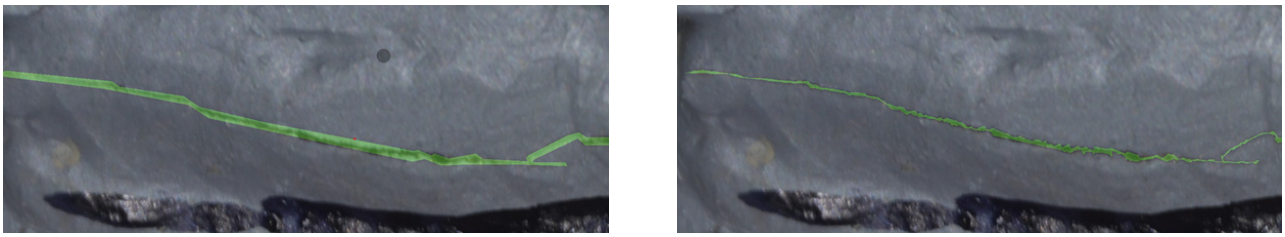
Fig. 1 The mainstream framework

of targets. Because object detection had been studied for a longer time than image segmentation, the performance of algorithms for deciding boxes of different instances is much better than that for classifying each pixel. In this context, a branch for mask prediction is added to a detection model, and this framework had been established to achieve exceptional results. Moreover, only binary classification is needed because images are cropped by the boundary of identified detections before being input into mask prediction. For example, Mask R-CNN (He *et al*. 2017) derived from Faster R-CNN only performs segmentation on proposals (i.e., region of interest), and the process is shown in Fig. 1(b). Eventually, numerous algorithms were proposed based on the aforementioned two frameworks to have better performance on public datasets.

For semantic segmentation, FCN suffers from that feature maps after being pooling multiple times would yield coarse output. To address this problem, a fusion of features at different layers was conducted by adding skip connections to obtain fine-grained information. For example, Jégou *et al*. (2017) proposed FC-DenseNet which introduced FCN into DenseNet to achieve better segmentation results and faster learning speed. The dense combination of features at different layers not only provided a better strategy for feature reuse but also reduced feature map explosion when upsampling. Meanwhile, another research focused on the improvement in the encoder-decoder structure. Badrinarayanan *et al*. (2017) presented SegNet where the decoder network was implemented with a hierarchy of decoders and by a corresponding encoder. Each decoder would use the max-pooling indices from the encoder. In that way, because only the indices should be stored instead of feature maps, SegNet was considered much more efficient. Although the problem resulted from the inevitable decrease in the resolution of feature maps caused by pooling layers, pooling was essential for enlarging the receptive field of each pixel to achieve high performance. In this context, dilated convolution was designed to substitute for pooling. The idea was first presented as the atrous convolution in DeepLab (Chen *et al*. 2017) and an extended structure, ASPP, was developed to address the issue of prediction on multi-scale targets. This DeepLab fundamentally resolved the shortcoming of pooling and also retained its advantages without increasing the number of parameters. Moreover, the model was maintained by the Google research team and rapidly improved in the last three years. Until now, three updates on the model were released to the public, and DeepLabv3+(Chen *et al*. 2018), was the latest one that implemented a decoder block in DeepLabv3 to fuse high-level and low-level features. DeepLabv3+ not only attains state-of-the-art performance on the public dataset but also is publicly available. Consequently, this study selects DeepLabv3+ to perform semantic segmentation on fatigue cracks on welding joints.

Because Mask R-CNN (He *et al*. 2017) successfully merged mask prediction and object detection through RoIAlign, this achievement completely changed the development direction of instance segmentation. Similarly, Zhou (2020) proposed an advanced version of You Only Look At Coefficients (YOLACT), YOLACT++, which reaches real-time processing with a comparable performance by adding a mask branch to a one-stage detector such as YOLOv3 (Redmon and Farhadi 2018). Since Faster R-CNN belongs to two-stage algorithms, one-stage detectors served as another option for faster inference and acceptable performance. Moreover, the detailed comparison on the performance of different models for instance segmentation had been elaborated by Zhou (2020), and Mask R-CNN was still rated to have higher performance. Two models which are Path Aggregation Network (PANet) and Mask Scoring R-CNN (MS R-CNN) outperformed Mask R-CNN. Instead of the substitution of the detector, both PANet and MS R-CNN focused on enriching features and developing better strategies for mask prediction. For PANet, a bottom-up path augmentation strengthened the connection between lower layers and the topmost feature. In addition, adaptive feature pooling was developed to aggregate features from all levels for each proposal. Thus, the performance is only higher than Mask R-CNN for about 4%. Nevertheless, the open-source was built on the older version of the framework, and the last update is made more than two years ago. For MS R-CNN, a MaskIoU head was designed to better score the instance segmentation hypothesis, and the mAP was improved by about 1%. This indicated a new direction to enhance instance segmentation. Despite the rise in the performance, Mask R-CNN still drew more community attention and had more stable open sources for extended applications. Therefore, this study also selects Mask R-CNN as the representation of instance segmentation and implements the model in experiments.

In summary, by reviewing research on deep learning-based image segmentation, two models, DeepLabv3+ and Mask R-CNN, are considered and implemented in experiments for identifying impacts of label quality on performance.



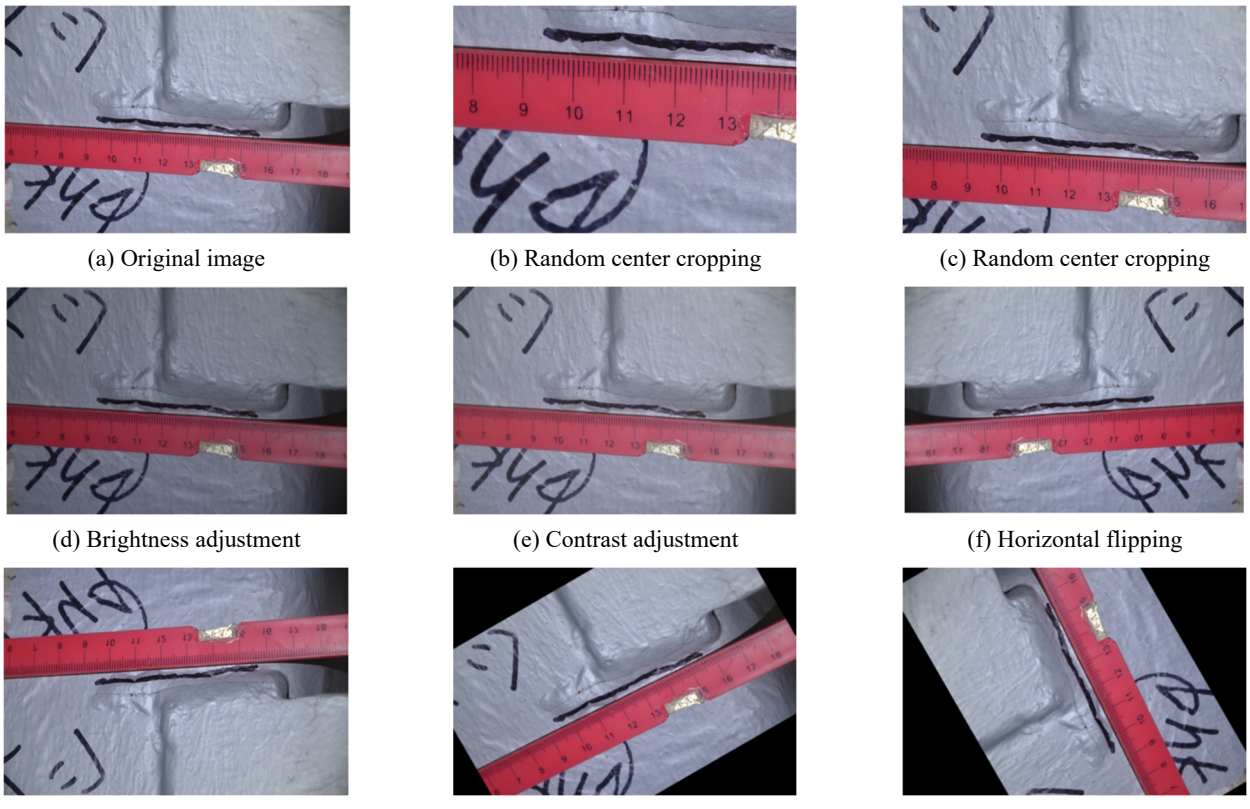
(a) By labelme (b) Refined by PixelAnnotation

Fig. 2 Demonstration of annotations

3. Method and dataset

3.1 Datasets for training and validation

As the performance of existing deep learning-based methods is data-driven, the composition of data should be elaborated, and the train-validation split ratio should be specified in different experiments. To begin with 200 images in the dataset, 120 images with well-annotation have two kinds of resolution, and among them, 19 images have a dimension of 5152 × 3864 while the rest are 4928 × 3264. Moreover, the resolution of all the other 80 unannotated images is 4928 × 3264, and these images are manually labeled by ourselves to enlarge the volume of training data. Because labeling policy determines what the model should learn from, variations in different images should be as little as possible. For those unannotated 80 images, only cracks located on welding joints or around the center of images are additionally labeled in this study. Under the assumption, two popular labeling tools were adopted to generate two different levels of label quality. For polygonal annotation, labelme (Wada 2016) provides a powerful interface for drawing labels through building critical points. For pixel-wise annotation, PixelAnnotationTool (Bréhéret 2017) where the painting brush is designed to draw labels is capable of delicate classification of pixels. When sketching the outline of cracks through labelme, background information may be accidentally included. In this context, the pixel-wise annotation is applied to further refining the label quality. The changes after refining are demonstrated in Fig. 2 and on average, the ratio of the area of annotations to the total area is decreased from 0.45% to 0.32%. Moreover, because the crack area is quite small, refining can introduce a significant reduction in the crack sizes. Then, the refined one can be sometimes up to 20 times smaller as the original one. The significant difference in the percentage between



(a) Original image (b) Random center cropping (c) Random center cropping

(d) Brightness adjustment (e) Contrast adjustment (f) Horizontal flipping

(g) Vertical flipping (h) Rotation (i) Rotation

Fig. 3 The result of data augmentation

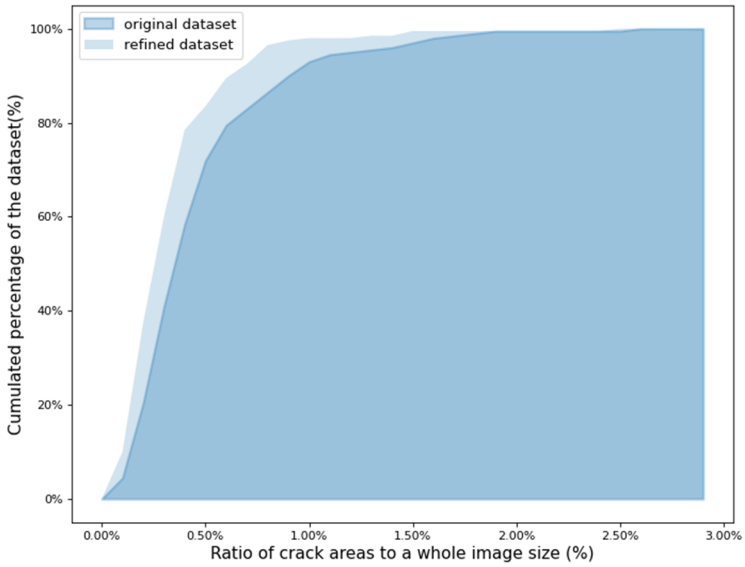


Fig. 4 The cumulative number of percentages of crack annotations

these two labeling policies results from the special characteristics of cracks. As most cracks are thin and long, more labor hours are needed to achieve more precise annotations. However, the relationship between label quality and performance of deep learning models is not clear at all. Therefore, this study carries out experiments to examine this uncertainty, and the result should provide researchers more effective strategies for data preparation. The annotations that are respectively generated by labelme and PixelAnnotation are named as follows.

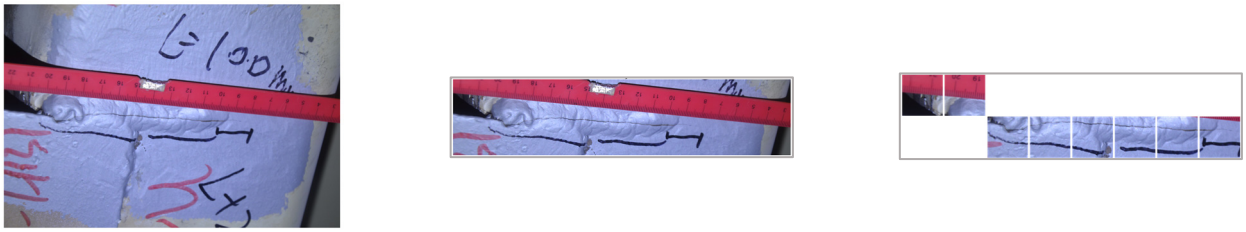
(1) *NLR* (*No Label Refining*): images with annotations by labelme.
(2) *LR* (*Label Refining*): images with annotations by PixelAnnotation.

Although the dataset has a total of 200 images with cracks, the number is still much smaller than public datasets such as the COCO dataset (Lin *et al*. 2014), which has more than 200K labeled images. A small dataset may suffer from overfitting and being vulnerable to unseen data. Consequently, data augmentation, which is a common approach to increase the data amount, is adopted to address the issue. In this paper, image processing including random center cropping, horizontal flipping, vertical flipping, rotation, etc. is used to augment the amount of data. Then, the data become 9 times bigger, and augmented images are shown in Fig. 3. Note that the data augmentation is only applied to training data. Moreover, the data are normalized by the mean and standard from other datasets, ImageNet (Deng *et al*. 2009), to expedite the learning process. This normalization is widely adopted in many studies because the volume of the dataset is very substantial (~1400k images). Additionally, these previous studies have shown certain improvements by normalization (Paszke *et al*. 2019).

Moreover, the small ratio of areas of cracks to the whole image may hinder model performance, and the cumulative number of percentages of crack annotations in each image is presented in Fig. 4. As can be seen, cracks in an image only account for a little percentage and most are even less than 1%. Additionally, many artificial marks or measuring tools on the surface have a similar appearance to cracks. The noise would confuse the model when identifying cracks; therefore, to alleviate the adverse effect of messy backgrounds, image cropping is employed to force images focusing on the annotations and containing less unnecessary information. Note that the experiments only remain the cropped images that contain cracks for further training. Likewise, the effectiveness of image cropping had been demonstrated in small object detection. Meng *et al*. (2017) introduced patch-level object detection by breaking the original image into small patches with fixed sizes. This innovation not only improved the performance of detecting small traffic signs but also make model learning more effective by reducing the use of memory. Furthermore, Ozge *et al*. (2019) proposed a tiling approach to enlarge the relative size of small objects to images under various tiling grid sizes, and the accuracy can be improved to three times bigger. Because of the accomplishments, Abdellatif *et al*. (2020) combined the block-based and pixel-based approaches for crack detection and took advantage of the former's noise robustness and the latter's pixel-wise localization. By determining if the block contains cracks, most false-positive predictions of segmentation can be precluded to enhance the accuracy. Through image cropping, the size of data can be augmented and the cracks are more closely centered in images with less background information to boost the model learning.

As illustrated in Fig. 5, this study implements three strategies for cropping as follows.

(1) *NC*: maintain the same size of images without any cropping.
(2) *CRec*: crop the original image into a minimum area that covers all crack annotations and slightly extends the area.
(3) *CBox*: crop the original image into couples of 512



(a) *NC* (b) *CRec* (a) *CBox*

Fig. 5 Demonstration of cropping strategies

× 512 rectangles and only keeps those containing crack annotations.

After the processing, the number of images in the *NC* and *CRec* remains the same while that in the *CBox* increases to 1076. Because the crack assessment in practice is usually developed based on large-scale images, the cropping strategies should be capable of determining a proper size of cracks in an image. To sum up, six different datasets with/without label refining and with different image cropping are employed for the training and validation. Finally, the above discussion elaborates on the composition of the dataset and image processing applied to the data. For attributes of the data, the areas of annotations are tiny, and a lot of noise has a similar appearance with cracks in the background. For image processing, data augmentation, normalization, and cropping are implemented to enhance the performance of segmentation models.

3.2 Experiment setup for models

First of all, based on the datasets, this paper conducted six experiments on the implemented models, DeepLabv3+ and Mask R-CNN, and each model corresponds to one of the six datasets. For the train-validation split ratio, the strategy where 80% for training and 20% for validation is commonly used in other papers, and even 70/30 is possible. By contrast, this paper set the ratio for training to 90% and that for validation to 10% because our dataset is not as large as other public datasets. Note that the split ratio in this paper is adopted to better training the models, and other split ratios are worth trying if more data are included. Thus, 180 images are randomly selected for training while 20 images are for validation. To prevent the unexpected influence of data variation, the division of the dataset in each experiment is the same and the detail is presented in Table 1. Note that the number of images has not yet been augmented, and the number should be 9 times bigger after data augmentation. In addition, the image cropping is applied to images after train-validation splitting. Meanwhile, the image cropping also results in more percentage of annotations. Subsequently, to evaluate model performance in different experiments, mean IoU (mIoU), which is widely utilized in semantic segmentation related papers, is adopted and serves as the same metric of identification accuracy in the competition. The mIoU is the average of each class IoU and expressed by

$$IoU = \frac{Intersection}{Union} = \frac{x \cap y}{x \cup y} \quad (1)$$

Table 1 The detail of adopted dataset of each experiment

Experiments	W/ image cropping	W/ label refining	Percentage of annotations (%)	training/ validation
NC_NLR	No	No	0.45	180/20
CBox_NLR	Yes	No	1.35	180/20
CRec_NLR	Yes	No	4.60	1,076/139
NC_LR	No	Yes	0.32	180/20
CBox_LR	Yes	Yes	0.92	180/20

where x is the area of prediction and y is the area of ground truth. After the experiments, the one with the highest mIoU will be submitted to the competition. Moreover, the impacts of label quality on model performance can be clarified, and the result should provide researchers with an empirical guideline on deciding label quality when preparing datasets.

In each experiment, the same model settings except backbones are assigned to eliminate the manmade uncertainties. Because the backbone of a CNN model that is responsible for extracting feature maps from raw images is independently determined in related papers, this paper discusses the most appropriate backbones for Mask R-CNN and DeepLabv3+. In this paper, the residual network (ResNet) (He *et al*. 2016), which is widely adopted and can achieve high performance on image segmentation, is selected as the backbone of DeepLabv3+ and Mask R-CNN. Moreover, ResNet proposed skip connections that can help gradient backpropagation, and this backbone is also named after the number of layers, e.g., ResNet-101, where the number, 101, is equal to the number of layers. In addition, Mask R-CNN is built on the algorithm for object detection. In the architecture, the fusion of multi-scale features is crucial to the detection performance, and thus an alternative backbone, namely R101-FPN which combines feature pyramid network (FPN) (Lin *et al*. 2017) with ResNet-101, is employed to fulfill the fusion. Then, both DeepLabv3+ and Mask R-CNN are built on a popular benchmark, Detectron2 (Wu *et al*. 2019), for vision tasks developed by Facebook AI Research (FAIR). In Detectron2, transfer learning is applied to all the two models to reduce the computational cost in training. For example, researchers spent 44 hours on 8 Nvidia Tesla M40 to obtain the final model, and even prototype testing can cost about a day in the original paper of Mask R-CNN (He *et al*. 2017). Consequently, transfer learning is preferable and adopted in this study. The pre-trained weights are downloaded from Detectron2 for DeepLabv3+ and Mask R-CNN and loaded

Table 2 Hyperparameters for training models

Hyperparameter	DeepLabv3+	Mask R-CNN
Batch size	8	4
Epoch/Iteration	200 epochs	150,000 iterations
Optimizer	stochastic gradient descent (momentum = 0.9, weight decay = 10^{-4})	
Learning rate	2×10^{-4} (poly decay)	
Epoch/Iteration	200 epochs	150,000 iterations

by the models before applying transfer learning.

The types and values of hyperparameters are presented in Table 2. Batch size corresponds to the size of a data sample per iteration, and gradients of each node are calculated through backpropagation to predict the samples and annotations. In this paper, batch sizes for training Mask R-CNN and DeepLabv3+ are different because of the memory limitation. Then, the optimizer updates weights in accordance with the gradients by a specific learning rate. For optimizer, this paper adopts the stochastic gradient descent (SGD) algorithm (Kiefer and Wolfowitz 1952) that is often used in training models such as DeepLabv3+ and Mask R-CNN. To prevent overfitting and better performance, the weight decay in the optimizer is set to restrain the update on weights while the polynomial decay in the learning rate helps models to converge. The feasibility of the above settings for training is examined in our previous work (Hsu *et al*. 2020), and the highest mIoU reached 68%. Additionally, other settings including loss functions unlisted above completely follow the studies for fair comparison (He *et al*. 2017, Chen *et al*. 2018).

Finally, six experiments are conducted on a single GPU, Nvidia Tesla V100 (Kirk 2007). The training duration of all experiments is less than two days. Although the two models use different representations of training times in terms of epoch and iteration, train convergences are still met in both models. The results of mask prediction are presented by the images in the validation set to demonstrate the impacts of label quality on the performance of DeepLabv3+ and Mask R-CNN models.

4. Result and discussion

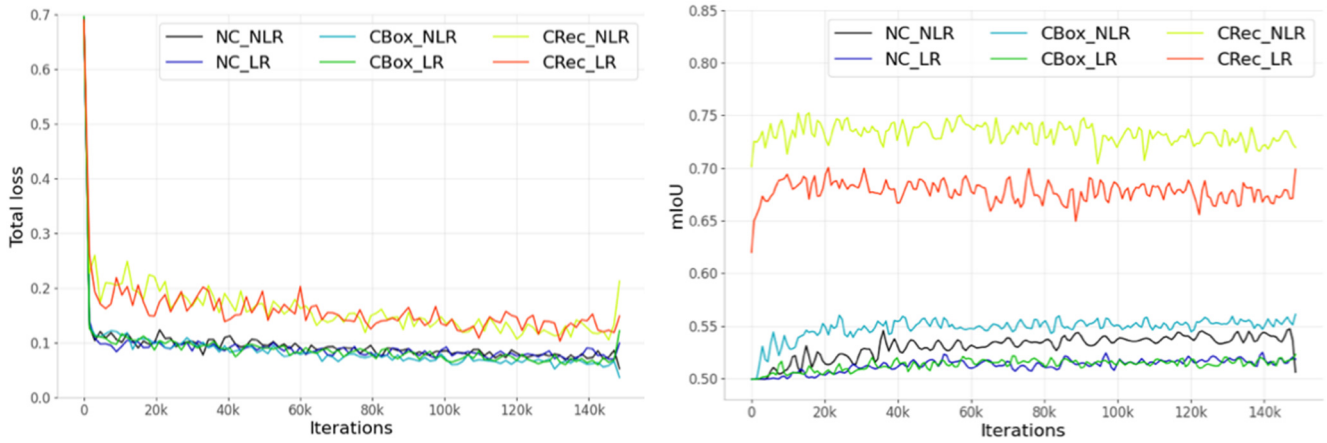
This section presents the experimental results on impacts of label quality on the performance of both DeepLabv3+ and Mask R-CNN models. First, the training losses and validation mIoUs are exhibited to understand training performance. Then, the comparison between the models that are trained by the datasets with original labels and refining labels is conducted to discuss impacts of label quality. Finally, the effect of label refining on DeepLabv3+ and Mask R-CNN models is indicated, and the most suitable labeling policy for each model is also presented.

4.1 Training and validation logs

The training and validation logs of DeepLabv3+ and Mask R-CNN are shown in Figs. 6 and 7, the originally labeled images (i.e., without label refining) are employed to calculate mIoUs. For the contents of the logs, training losses are an important indicator of model convergence and the lower the losses are, the better the model learns. Because losses that are calculated by numeric outputs including pixel classification can partially indicate the quality of model predictions, mIoU as previously mentioned can better present the completeness of predictions than losses. In a word, this paper uses training losses to explain how the models are trained and validation mIoUs to represent the models' performance. As can be seen in Fig. 6(a) and Fig. 7(a), the losses quickly drop in the beginning, that is, transfer learning loads pre-trained weights and yields models to converge much faster.

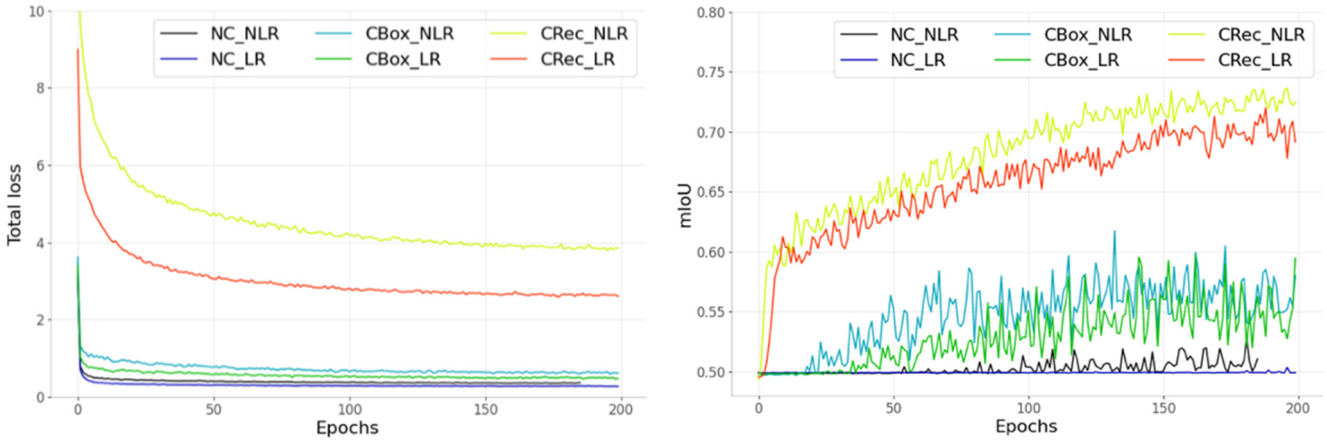
Moreover, Figs. 6(a) and 7(a) show that smaller cropping areas introduce larger losses. *CRec_NLR* and *CRec_LR* have a little smoother curve before achieving convergence. Although the losses are increased after image cropping, the validation mIoUs have significant improvement as shown in Figs. 6(b) and 7(b). The highest mIoU is around 0.75 in both DeepLabv3+ and Mask R-CNN. By contrast, the mIoUs of *NC_NLR* and *NC_LR* are almost equal to 0.5 which is a very low value of the metric. These low mIoUs may be introduced by the background confusion because cracks only occupy a tiny part of the images shown in Table 1. Thus, the extremely low losses should presumably result from overfitting to backgrounds. Note that most image backgrounds contain multiple hand-writing markers. Image cropping can enhance the percentage of annotations and thus help better model learning. As a result, the performance of the model with image cropping via size 512 × 512 outperforms the others.

The effects of label refining in model training are then discussed. In general, the images with higher label quality should result in more accurate predictions because models can be trained by more precise annotations. However, as seen in mIoUs in both Figs. 6(b) and 7(b), label refining reduces mIoUs in both DeepLabv3+ and Mask R-CNN with the same image cropping strategy. Moreover, when the models with label refining are applied to the validation dataset, the higher label quality also indicates more rigorous evaluation and may yield lower mIoUs. To better represent the effectiveness, the trained models are additionally validated by the images with/without label refining, and the corresponding mIoUs are calculated and listed in Table 3. As found in the results, the same model indeed has poorer performance when being evaluated by a validation dataset with label refining. For Mask R-CNN, all mIoUs decrease. The difference becomes bigger if image cropping by a smaller area is adopted. Because an increase in label quality depends on manual elimination of abundant annotations, rough borders of annotations and even fragmented annotations are induced. Unlike DeepLabv3+, Mask R-CNN needs not only the mask annotations but also the geometry of bounding boxes. Moreover, this study simply generated a minimum bounding box that can cover all mask annotations as an instance for Mask R-CNN. Thus, one crack may have multiple fragmented annotations and result in different instances. Consequently, algorithms of instance segmentation should perform better on datasets with



(a) Training loss (b) Validation mIoU

Fig. 6 Training and validation logs of Mask R-CNN



(a) Training loss (b) Validation mIoU

Fig. 7 Training and validation logs of DeepLabv3+

Table 3 mIoUs evaluated by validation datasets with and without label refining

Validation dataset		Training dataset mIoU (IoU-background/IoU-crack)					
		NC_NLR	*CBox_NLR*	*CRec_NLR*	*NC_LR*	*CBox_LR*	*CRec_LR*
Evaluated annotations with label refining	DeepLabv3+	0.4999 (0.9989/0.0009)	0.5854 (0.9943/0.1765)	0.6473 (0.9836/0.3111)	0.4994 (0.9989/0.0000)	0.6100 (0.9979/0.2221)	0.6736 (0.9878/0.3594)
	Mask R-CNN	0.5324 (0.9982/0.0666)	0.5413 (0.9973/0.0853)	0.6890 (0.9903/0.3877)	0.5117 (0.9987/0.0247)	0.5206 (0.9979/0.0433)	0.6856 (0.9917/0.3796)
Evaluated annotations without label refining	DeepLabv3+	0.4995 (0.9984/0.0007)	0.6112 (0.9945/0.2279)	0.6942 (0.9850/0.4034)	0.4992 (0.9984/0.0000)	0.6126 (0.9975/0.2277)	0.7108 (0.9882/0.4335)
	Mask R-CNN	0.5360 (0.9978/0.0742)	0.5554 (0.9968/0.1139)	0.7201 (0.9901/0.4500)	0.5117 (0.9982/0.0251)	0.5200 (0.9970/0.0429)	0.6906 (0.9901/0.3911)

polygon-based annotations, which have a much smoother border and prevent mistaking an instance for multiple instances. In short, evaluating crack detection from both DeepLabv3+ and Mask R-CNN by datasets with different label quality may misrepresent real performance because for the same model, the increase in label quality of the validation dataset can cause lower mIoUs.

4.2 Effects of annotation strategies on model performance

To investigate the influence of label quality, both image cropping and label refining are respectively discussed directly from the images. First, Fig. 8 shows the detection results from both DeepLabv3+ and Mask R-CNN with

(a) Original image

(b) Annotations

mIoU = 0.5
(IoU-crack = 0)
(c) DeepLabv3+ with *NC_NLR*

mIoU = 0.551
(IoU-crack = 0.105)
(d) DeepLabv3+ with *CBox_NLR*

mIoU =0.734
(IoU-crack = 0.476)
(e) DeepLabv3+ with *CRec_NLR*

mIoU = 0.551
(IoU-crack = 0.105)
(f) Mask R-CNN with *NC_NLR*

mIoU = 0.549
(IoU-crack = 0.107)
(g) Mask R-CNN with *CBox_NLR*

mIoU = 0.665
(IoU-crack = 0.343)
(h) Mask R-CNN with *CRec_NLR*

Fig. 8 Predictions by DeepLabv3+ and Mask R-CNN with different strategies for image cropping and without label refining

image cropping strategies. As seen in this figure, more pixels are successfully identified after image cropping was employed, and the effect of *CRec* is the most significant.

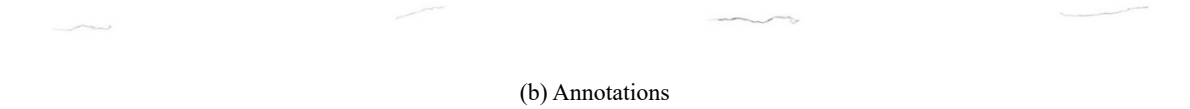
For semantic segmentation, DeepLabv3+ cannot even identify any cracks using NC, and the mIoU is equal to 0.5. After applying image cropping, the model performance is greatly improved. As presented in Figs. 8(d) and 8(e), CBox and *CRec* result in a mIoU of 0.55 and 0.73. Despite the false predictions of cracks on artificial marks, CBox enforces DeepLabv3+ to be focused on a specific area of an image and then helps DeepLabv3+ learn crack-like features.

In this context, *CRec* with a smaller cropping size induces a more notable improvement than *CBox* by reducing more background noise, and DeepLabv3+ can better distinguish cracks from artificial marks using *CRec*. For instance segmentation, Mask R-CNN can still identify parts of cracks using NC and has a mIoU of 0.55 that is better than DeepLabv3+. Similarly, *CRec* also greatly improves the model performance, and Mask R-CNN achieves a higher mIoU using *CRec*. On the contrary, *CBox* has a negative effect on Mask R-CNN without improving the model performance. As seen in Fig. 8(g), Mask R-CNN predicts a more complete crack in the middle of the image using *CBox*. Still, this model fails to identify the crack in the corner. To sum up, image cropping aids models to learn more detailed crack features. Moreover, *CRec* is quite effective to enhance both DeepLabv3+ and Mask R-CNN performance.

The image cropping effects are discussed by both DeepLabv3+ and Mask R-CNN predictions on the validation and test dataset. Fig. 9 displays another set of detection result when the image cropping strategies are applied. In Figs. 9(d) and 9(g), *CRec* improves both DeepLabv3+ and Mask R-CNN because mIoUs increase by 10%. Although more pixels of cracks are identified by learnt crack features from more specific areas, *CBox* has a distinct effect on Mask R-CNN from that on DeepLabv3+.

For Mask R-CNN, segmentation is reached through couples of reliable proposals, and the improvement in the model performance after applying image cropping is not as huge as that of DeepLabv3+. In Fig. 9(e), Mask R-CNN is still capable of identifying a large part of crack pixels via NC, and performance of trained models from NC is even better that from *CBox*. Fig. 10 demonstrates that the trained models from NC and *CBox* can successfully predict bounding boxes of cracks. Thus, Mask R-CNN is not quite beneficial from the image cropping of *CBox*. Although accurate bounding boxes are provided in the training dataset, the squeezed areas can still fail in segmentation of Mask R-CNN. However, the same image cropping attains a fair improvement in DeepLabv3+. Instead, *CRec* employs a much smaller cropping size than predicted boxes. The Mask R-CNN model trained from the datasets surpasses the others, and the performance is comparable with DeepLabv3+. Because Mask R-CNN performs multi-task learning in which segmentation is considered as an additional branch to object detection and DeepLabv3+ is only designed for image segmentation, the model complexity of Mask R-CNN for segmentation is much lower than that of DeepLabv3+. Therefore, a better

(a) Original image



(b) Annotations

mIoU = 0.520 (IoU-crack = 0.049) | mIoU = 0.532 (IoU-crack = 0.069) | mIoU = 0.596 (IoU-crack = 0.198) | mIoU = 0.534 (IoU-crack = 0.077)

(c) DeepLabv3+ with *CBox_NLR*

mIoU = 0.690 (IoU-crack = 0.388) | mIoU = 0.627 (IoU-crack = 0.263) | mIoU = 0.699 (IoU-crack = 0.412) | mIoU = 0.679 (IoU-crack = 0.367)

(d) DeepLabv3+ with *CRec_NLR*

mIoU = 0.622 (IoU-crack = 0.244) | mIoU = 0.526 (IoU-crack = 0.053) | mIoU = 0.627 (IoU-crack = 0.257) | mIoU = 0.576 (IoU-crack = 0.154)

(e) Mask R-CNN with *NC_NLR*

mIoU = 0.637 (IoU-crack = 0.282) | mIoU = 0.570 (IoU-crack = 0.147) | mIoU = 0.580 (IoU-crack = 0.178) | mIoU = 0.543 (IoU-crack = 0.094)

(f) Mask R-CNN with *CBox_NLR*

mIoU = 0.591 (IoU-crack = 0.199) | mIoU = 0.594 (IoU-crack = 0.206) | mIoU = 0.712 (IoU-crack = 0.276) | mIoU = 0.629 (IoU-crack = 0.447)

(g) Mask R-CNN with *CRec_NLR*

Fig. 9 Demonstration of model predictions on the test dataset with different image cropping strategies and without label refining

cropping strategy is to precisely segment cracks from a smaller area for Mask R-CNN.

Even though label refining decreases the percentage of annotations, the quality of annotations is improved. In Fig. 11, some pixels of cracks fail to be segmented, i.e., results in (c) and (f); however, label refining results in a worse trained model to predict a finer level of cracks, i.e., results in (d) and (g). After applying label refining, both DeepLabv3+ and Mask R-CNN identify less pixels of cracks, and corresponding mIoUs decrease. As seen in the experimental results, annotations that slightly exceed the actual crack areas are acceptable and can be somehow beneficial to model training, especially for tiny cracks. Label refining can improve the overall performance of DeepLabv3+ when the number of tiny cracks is lower. Thus, a trade-off exists between exceeding areas of annotations and model performance and is worth exploring in the future.

(a) Mask R-CNN with *NC_NLR* (b) Mask R-CNN+ with *CBox_NLR*

Fig. 10 Visualization of both bounding boxes and segmentation predicted by Mask R-CNN

Table 4 Highest mIoUs of DeepLabv3+ and Mask R-CNN in the whole training

Training dataset	DeepLabv3+	Mask R-CNN
NC_NLR	0.5274	0.5460
CBox_NLR	0.6175	0.5612
CRec_NLR	0.7363	0.7546
NC_LR	0.5036	0.5232
CBox_LR	0.5979	0.5256
CRec_LR	0.7200	0.7016

As listed in Table 4, Mask R-CNN, which is trained by the dataset with image cropping into rectangles of size 512 × 512 (*CRec*) and no label refining (*NLR*), achieves the highest mIoU among all models. The finely tuned model, which has highest crack detection accuracy, is available from Google Colab notebook[1]. Because label refining may have a negative effect on training models, in particular of Mask R-CNN to segment tiny cracks, these cases can be a vital reference for future researchers when labeling their own datasets. To further clarify the influence of refining annotated areas on model performance, Mask R-CNN with *CRec_NLR* and with *CRec_LR* are evaluated on both training and validation images one by one, and the results are demonstrated in Fig. 12. Each image should have two kinds of annotations, namely *LR* and *NLR* which separately correspond to annotations with and without label refining. Using the two annotations, three different comparisons are carried out as follows.

(1) *IoU-crack*(*LR-LR*): the crack IoU of Mask R-CNN with *CRec_NLR* and *CRec_LR* that are both evaluated by images with *LR*.
(2) *IoU-crack*(*NLR-LR*): the crack IoU of Mask R-CNN with *CRec_NLR* and *CRec_LR* that are evaluated by images with *NLR* and *LR*, respectively.
(3) *IoU-crack*(*NLR-NLR*): the crack IoU of Mask R-CNN with *CRec_NLR* and *CRec_LR* that are both evaluated by images with *NLR*.

Instead of using mIoU, only the IoU of crack is presented to exclude background influence from model performance on segmenting cracks. The comparisons, except for *IoU-crack*(*NLR-LR*), are generated from the same type of annotations, and *IoU-crack*(*NLR-LR*) indicates the results of the experiments where models are evaluated by their own validation dataset. In Fig. 12, the y-coordinate represents the IoU difference of cracks obtained by subtracting cracks IoU of *CRec_NLR* from that of *CRec_LR*. For the x-coordinate, Figs. 12(a) and (c) employs the value by subtracting the ratio of annotations with label refining from that without label refining, and Figs. 12(b) and (d) exploit the value of annotations with label refining. Because label refining is to precisely locate the area of cracks, the value of annotations with label refining should represent the actual area of cracks in an image. Therefore, Figs. 12(b) and (d) show the effects of crack sizes on the impacts of label quality to model performance.

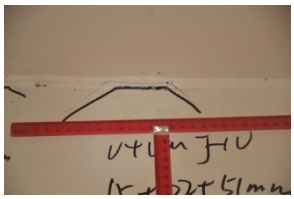
As seen Figs. 12(a) and (c), the performance difference becomes largely negative, resulting in a worse model to be obtained after label refining. Despite the case of *IoU-crack* (*LR-LR*), *CRec_NLR* can achieve comparable performance to *CRec_LR*. Moreover, *CRec_NLR* is trained by lower-quality annotations. Still, *CRec_NLR* outperforms *CRec_LR* as shown in Fig. 12(a). When the ratio difference after label refining becomes larger, model performance significantly decreases. The evaluation using the validation dataset also shows that *CRec_NLR* outperforms *CRec_LR* in Fig. 12(c), but the decrease along with ratio difference is almost the same.

To explore the worse cases using label refining, Figs. 12(b) and (d) demonstrate the relationship between the annotated areas after label refining and performance differences. As can be seen, the slope of curves is positive while a large amount of the points has negative changes in model performance. Thus, the models after label refining barely recognize cracks being lower than a certain area. Additionally, Fig. 12(b) shows the dataset with a large number of small cracks. As seen in Table 4.

4.3 Empirical guideline on data preparation

Although data preparation costs considerable time, especially for the generation of annotations for deep learning crack segmentation, previous studies mainly elaborated model establishment without discussing label policies and criteria (Song *et al*. 2019, Kalfarisi *et al*. 2020). As seen, most researchers employ their intuition and subjective view to label the images, and perhaps limited time was spent on repeatedly checking the annotations. Therefore, an empirical guideline on data preparation concerning image cropping and label refining is provided and discussed in this section. The images released after the contest are also utilized to demonstrate the differences

[1]https://colab.research.google.com/drive/1Xbo7cQdoHPgEX73QVAfjBXGFrQYW_xnH?usp=sharing



(a) Original image

(b) Annotations

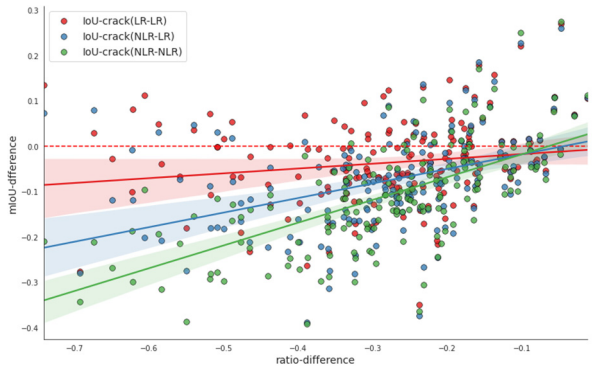
mIoU = 0.644
(IoU-crack = 0.295)
(c) DeepLabv3+ with *CRec_NLR*

mIoU = 0.625
(IoU-crack = 0.255)
(d) DeepLabv3+ with *CRec_LR*

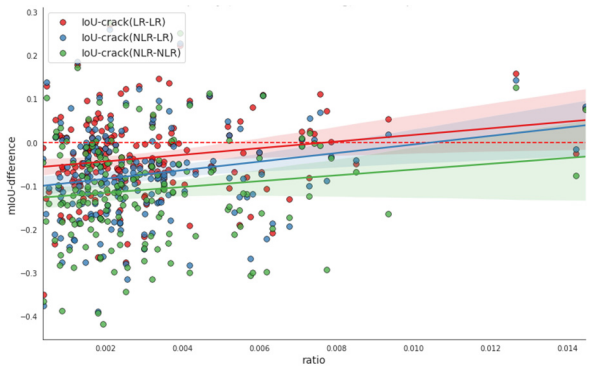
mIoU = 0.692
(IoU-crack = 0.392)
(f) Mask R-CNN with *CRec_NLR*

mIoU = 0.678
(IoU-crack = 0.368)
(g) Mask R-CNN with *CRec_LR*

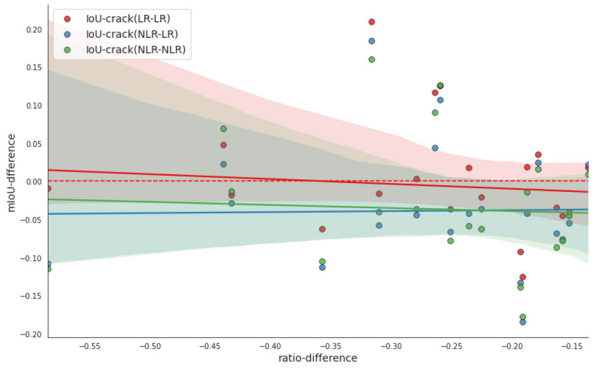
Fig. 11 Effects of label refining for both DeepLabv3+ and Mask R-CNN



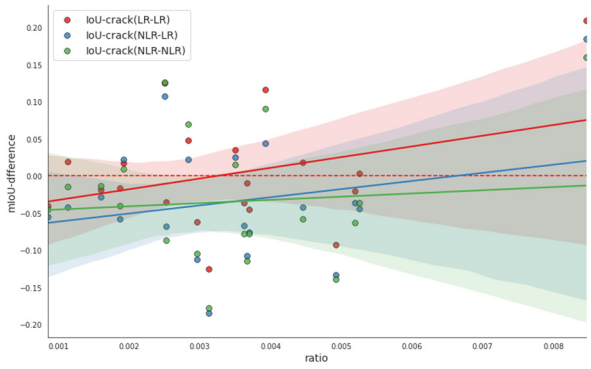
(a) Ratio difference after label refining and model performance evaluated by training images



(b) Ratios after label refining and model performance evaluated by training images



(c) Ratio difference after label refining and model performance evaluated by validation images



(d) Ratios after label refining and model performance evaluated by validation images

Fig. 12 Relationship between the ratio of annotated areas in an image and model performance (mIoU)

among the model predictions in various experiments. The model prediction results on the dataset are available from Google Drive[2].

First, Table 3 shows that the improvements by image cropping are both significant for DeepLabv3+ and Mask R-CNN. As found in the prediction results, the circumstances that these two segmentation models should be applied with image cropping are described in the following.

[2] https://drive.google.com/drive/folders/1Ha28ka3j1SxB-aDL4WaXhBUCGpHRVCA8?usp=sharing

- **Semantic segmentation.** Because of the zoom-in effect, image cropping can greatly enhance the performance of semantic segmentation by reducing the background noise. DeepLabv3+ fails to identify any crack using *NC* as seen in Fig. 8(c) while DeepLabv3+ can recognize crack-like objects including artificial marks using *CBox* as seen in Fig. 8(d). Moreover, Fig. 8(e) shows that *CRec* with a smaller cropping size lowers most false predictions on crack-like objects by DeepLabv3+. Nevertheless, the smaller cropping size can introduce more computational cost. Determining a proper size is vital to prevent an excessive number of training images.
- **Instance segmentation.** Both Figs. 8(g) and 9(f) shows that mIoUs decrease after applying *CBox*. This is because most instance segmentations follow the framework of multi-task learning including object detection that has a similar function to image cropping. Therefore, the cropping size should be at least smaller than the bounding boxes of objects to exert the effect of reducing background noise.

Subsequently, label refining that enforces annotations less exceeding actual cracks transforms polygon-based annotations to pixel-based annotations. The processing has an inversed effect on these two models. In general, polygon-based annotations require fewer labor hours and are considered to be sufficient for Mask R-CNN, and label refining induces difficulties to recognize tiny cracks by models. In contrast, pixel-based annotations need more labor hours and improve the performance of DeepLabv3+. As shown in Fig. 12 (b) and (d), a crack whose area occupies less than 0.3% of an image is likely to decline in performance. Thus, such cracks can be considered tiny cracks, and label refining may be less effective to train better models. The followings provide the suggestions when labeling image data for similar cracks.

- **Semantic segmentation.** Because label refining provides a more distinct demarcation between pixels of cracks and contiguous background, semantic segmentation models trained by less confusing scenarios can achieve a more accurate classification of each pixel of an image. As the result in Table 3, the mIoU can be improved by around 3%. However, Fig. 11(d) demonstrates that DeepLabv3+ fails to predict partially tiny cracks, and label refining results in decreasing mIoUs for such case. If the dataset consists of more tiny cracks than large cracks, image data should be labeled by polygon-based annotations.
- **Instance segmentation.** Because the branch of segmentation in instance segmentation is usually much simpler than semantic segmentation, the appearance information about tiny cracks is prone to losing during feature extraction. Similarly, Fig. 11(g) shows that Mask R-CNN also suffers from finer annotations when predicting tiny cracks. Moreover, label refining can yield fragmented cracks that confuse Mask R-CNN by mistaking a single crack for multiple cracks. Consequently, instance segmentation is recommended to use polygon-based annotations when preparing a dataset. Even if pixel-based annotations are available in the beginning, slightly enlarging the annotations should be beneficial to the accuracy of crack identification.

Finally, the empirical guideline on the data preparation aims to strengthen identification successfulness via image cropping and label refining. Without any processing on training images, DeepLabv3+ and Mask R-CNN can only achieve a mIoU of 50%, and the results indicate that models barely identify any cracks in an image. After the utilization of image cropping and label refining, the mIoU can increase up to 70%. Therefore, labeling policies can significantly influence prediction accuracy and performance for both DeepLabv3+ and Mask R-CNN.

5. Conclusions

This paper aimed to explore the impacts of label quality on model performance and to provide the most appropriate labeling strategy for training models. Two state-of-the-art deep learning-based models including Mask R-CNN and DeepLabv3+ were evaluated and implemented, and two datasets with and without label refining were prepared. Six experiments based on various datasets were carried out to train the two models for crack identification as well as to explore the impacts of label quality on performance. As seen in the results, the highest mIoU of 75.46% was achieved by Mask R-CNN when training by the dataset with image cropping into rectangles of size 512 × 512 and without label refining. Moreover, the impacts of label quality on these two models were polarized. Annotations at a finer level confused Mask R-CNN and resulted in decreased performance. The finer annotations improved the precision of mask predictions by DeepLabv3+. Moreover, the impacts were associated with types of cracks. For tiny cracks, labeling with slight exceedance of actual areas can yield models with better learning. As found from the results, an empirical guideline was presented to provide more effective strategies for labeling cracks in various cases. Because a trade-off existed between exceeding areas of annotations and model performance, the relationship should be further studied to effectively work on labeling data. The research on developing deep learning models for crack segmentation should also elaborate the label quality of the dataset for evaluating the methods.

Acknowledgments

The authors would like to thank the organizations of the International Project Competition for SHM (IPC-SHM 2020) ANCRiSST, Harbin Institute of Technology (China), and University of Illinois at Urbana-Champaign (USA) for their generosity of providing the invaluable data. The authors also would like to thank the chairs of IPC-SHM 2020 Prof. Hui Li, and Prof. Billie F. Spencer Jr. for their leadership on the competition.

References

Abdellatif, M., Peel, H., Cohn, A.G. and Fuentes, R. (2020), "Combining block-based and pixel-based approaches to improve crack detection and localization", *Automat. Constr.*, **122**, 103492. https://doi.org/10.1016/j.autcon.2020.103492

Albrecht, P. and Wright, W. (2000), "Fatigue and fracture of steel bridges", *Eur. Struct. Integr. Soc.*, **26**, 211-234. https://doi.org/10.1016/S1566-1369(00)80051-5

Badrinarayanan, V., Kendall, A. and Cipolla, R. (2017), "Segnet: a deep convolutional encoder-decoder architecture for image segmentation", *IEEE Transact. Pattern Anal. Mach. Intell.*, **39**(12), 2481-2495. https://doi.org/10.1109/TPAMI.2016.2644615

Bréhéret, A. (2017), "Pixel Annotation Tool", Retrieved from: https://github.com/abreheret/PixelAnnotationTool

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K. and Yuille, A.L. (2017), "Deeplab: semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs", *IEEE Transact. Pattern Anal. Mach. Intell.*, **40**(4), 834-848. https://doi.org/10.1109/TPAMI.2017.2699184

Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F. and Adam, H. (2018), "Encoder-decoder with atrous separable convolution for semantic image segmentation", *Proceedings of the European Conference on Computer Vision*, Munich, Germany, September.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K. and Fei-Fei, L. (2009), "Imagenet: a large-scale hierarchical image database", *IEEE Conference on Computer Vision and Pattern Recognition*, Miami, FL, USA, June.

Dung, C.V. (2019), "Autonomous concrete crack detection using deep fully convolutional neural network", *Automat. Constr.*, **99**, 52-58. https://doi.org/10.1016/j.autcon.2018.11.028

Farrar, C.R. and Worden, K. (2007), "An introduction to structural health monitoring", *Philos. Transact. Royal Soc. A: Mathe. Phys. Eng. Sci.*, **365**(1851), 303-315. https://doi.org/10.1098/rsta.2006.1928

Feng, D. and Feng, M.Q. (2018), "Computer vision for SHM of civil infrastructure: From dynamic response measurement to damage detection–A review", *Eng. Struct.*, **156**, 105-117. https://doi.org/10.1016/j.engstruct.2017.11.018

He, K., Zhang, X., Ren, S. and Sun, J. (2016), "Deep residual learning for image recognition", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, NA, USA, June.

He, K., Gkioxari, G., Dollár, P. and Girshick, R. (2017), "Mask r-cnn", *Proceedings of the IEEE International Conference on Computer Vision*, Venice, Italy, October.

Hsu, S.-H., Chang, T.-W. and Chang, C.-M. (2020), "Concrete Surface Crack Segmentation Based on Deep Learning", In: *European Workshop on Structural Health Monitoring* (Lecture Notes in Civil Engineering), Vol. 128, pp. 24-34. https://doi.org/10.1007/978-3-030-64908-1_3

Jégou, S., Drozdzal, M., Vazquez, D., Romero, A. and Bengio, Y. (2017), "The one hundred layers tiramisu: fully convolutional densenets for semantic segmentation", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, Honolulu, HI, USA, July.

Kalfarisi, R., Wu, Z.Y. and Soh, K. (2020), "Crack detection and segmentation using deep learning with 3D reality mesh model for quantitative assessment and integrated visualization", *J. Comput. Civil Eng.*, **34**(3), 04020010. https://doi.org/10.1061/(ASCE)CP.1943-5487.0000890

Kiefer, J. and Wolfowitz, J. (1952), "Stochastic estimation of the maximum of a regression function", *Math. Statist.*, **23**(3), 462-466.

Kirk, D. (2007), "NVIDIA CUDA software and GPU parallel computing architecture", *Proceedings of the 6th International Symposium on Memory Management*, New York, NY, USA, October.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P. and Zitnick, C.L. (2014), "Microsoft coco: common objects in context", *Proceedings of the European Conference on Computer Vision*, Zurich, Switzerland, September.

Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B. and Belongie, S. (2017), "Feature pyramid networks for object detection", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Honolulu, HI, USA, July.

Long, J., Shelhamer, E. and Darrell, T. (2015), "Fully convolutional networks for semantic segmentation", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Boston, MA, USA, June.

Meng, Z., Fan, X., Chen, X., Chen, M. and Tong, Y. (2017), "Detecting small signs from large images", *IEEE International Conference on Information Reuse and Integration*, San Diego, CA, USA, August.

Mohan, A. and Poobal, S. (2018), "Crack detection using image processing: a critical review and analysis", *Alexandria Eng. J.*, **57**(2), 787-798. https://doi.org/10.1016/j.aej.2017.01.020

Ozge Unel, F., Ozkalayci, B.O. and Cigla, C. (2019), "The power of tiling for small object detection", *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, Long Beach, CA, USA, June.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L. and Desmaison, A. (2019), "PyTorch: an imperative style, high-performance deep learning library", *Adv. Neural Inform. Process. Syst.*, **32**, 8024-8035.

Redmon, J. and Farhadi, A. (2018), "Yolov3: an incremental improvement", arXiv preprint, arXiv: 1804.02767.

Ren, S., He, K., Girshick, R. and Sun, J. (2015), "Faster r-cnn: towards real-time object detection with region proposal networks", *IEEE Transact. Pattern Anal. Mach. Intell.*, **39**(6), 1137-1149. https://doi.org/10.1109/TPAMI.2016.2577031

Song, G., Wang, C. and Wang, B. (2017), "Structural health monitoring (SHM) of civil structures", *Appl. Sci.*, **7**(8), 789. https://doi.org/10.3390/app7080789

Song, Q., Wu, Y., Xin, X., Yang, L., Yang, M., Chen, H., Liu, C., Hu, M., Chai, X. and Li, J. (2019), "Real-time tunnel crack analysis system via deep learning", *IEEE Access*, **7**, 64186-64197. https://doi.org/10.1109/ACCESS.2019.2916330

Wada, K. (2016), "labelme: Image Polygonal Annotation with Python", Retrieved from: https://github.com/wkentaro/labelme

Wu, Y., Kirillov, A., Massa, F., Lo, W.-Y. and Girshick, R. (2019), "Detectron2", Retrieved from: https://github.com/facebookresearch/detectron2

Zhou, C. (2020), "Yolact++ better real-time instance segmentation", Ph.D. Dissertation; University of California, Davis, CA, USA.