Automation in Construction 110 (2020) 103018

Contents lists available at ScienceDirect

Automation in Construction

journal homepage: www.elsevier.com/locate/autcon

Densely connected deep neural network considering connectivity of pixels for automatic crack detection

Qipei Mei, Mustafa Gül*, Md Riasat Azim

Department of Civil and Environmental Engineering, University of Alberta, Edmonton, Alberta T6G 2W2, Canada

ARTICLE INFO

Keywords:
Crack detection
Deep learning
Transposed convolution layer
Densely connected layers
Connectivity of pixels

ABSTRACT

In order to develop smart cities, the demand for assessing the condition of existing infrastructure systems in an automated manner is burgeoning rapidly. Among all the early signs of potential damage in infrastructure systems, formation of cracks is a critical one because it is directly related to the structural capacity and could significantly affect the serviceability of the infrastructure. This paper proposes a novel deep learning-based method considering the connectivity of pixels for automatic pavement crack detection which has the potential to complement the current practice involving visual inspection which is costly, inefficient and time-consuming. In the proposed method, the convolutional layers are densely connected in a feed-forward fashion to reuse features from multiple layers, and transposed convolution layers are used for multiple level feature fusion. A novel loss function considering the connectivity of pixels is introduced to overcome the issues related to the output of transposed convolution layers. The proposed method is tested on two datasets, where the first one is collected from a handheld smartphone and the second one is collected from a high-speed camera mounted on the rear of a moving car. In both datasets, the proposed method shows superior performance than other available methods.

1. Introduction

The development of smart infrastructure systems to provide residents with high levels of comfort is imperative for the modern age. One crucial component of smart infrastructure systems is timely sensing and monitoring to ensure the safety of the infrastructures as well as the people who use them. Existing infrastructure systems are experiencing a variety of risks due to overloading and environmental effects in their life cycles [1]. It is imperative to identify potential damage in its early stage, by which not only severe accidents can be prevented but also costs can be reduced. A number of researchers in recent years have shown interest in developing crack detection mechanisms to provide early signs of potential damage in building materials [2–5]. The current common practice for crack detection is primarily based on visual inspection which is costly and inefficient.

To resolve the issues mentioned above and to automate the crack detection process, researchers have developed methodologies based on different techniques, such as vibration signals, ultrasound, laser or images [6–11]. For the cracks on the surface of building materials, image-based crack detection methods are among the most promising ones because cameras are more accessible than other tools these days especially after the widespread of digital cameras and smartphones [12].

Generally, there are two branches in the field of image-based crack detection studies. The first one is based on traditional image processing techniques. The researchers handcrafted a series of filters to distinguish the cracks from background or noise [13–16]. Since cracks are sometimes very similar to the texture of the material surfaces, it is challenging to distinguish the cracks from the background. Abdel-Qader et al. [17] applied and compared four basic edge detection methods on images and determined that Haar transform outperformed other methods in terms of crack detection. Sinha et al. [15] developed a method to detect cracks in buried pipe images which first extracted local features from the pixels and then to find cracks among segment candidates through cleaning and linking. In the method proposed by Fujita and Hamamoto [16], they subtracted the median filtered image from the original one to expose the location of cracks on the concrete surface and then applied a line filter based on Hessian matrix and probabilistic relaxation algorithm to further highlight and link the cracks. According to Iyer and Sinha [18], cracks in buried pipes are darkest in the images, locally linear and have tree-like geometry. Based on this observation, they presented a 3-step method based on mathematical morphology and curvature evaluation for crack detection in a noisy environment. Many more relevant studies can be found in [18–21].

Despite significant progress made in image processing based crack

* Corresponding author.
E-mail address: mustafa.gul@ualberta.ca (M. Gül).

https://doi.org/10.1016/j.autcon.2019.103018
Received 20 June 2019; Received in revised form 20 October 2019; Accepted 16 November 2019
Available online 26 November 2019

detection methods, they have their limitations. One of the biggest challenges is that the features are usually manually designed for some crack detection tasks, and they may not work properly if some conditions are changed. In real-life applications, it is almost impossible for one method to work for all the cases due to the irregularity of cracks, the complex illumination conditions and the varying texture in material surfaces [22].

Therefore, researchers tend to seek solutions in another branch, i.e. machine learning-based methods. Among all machine learning techniques, the methods using deep learning have shown superior performance on crack detection tasks. Deep learning-based methods have significant advantages compared to traditional image processing techniques or other machine learning techniques [22–25]. First, there is no need to manually design features to distinguish cracks from the background. The features are learned automatically from the training process. Second, the deep neural networks can implicitly consider the complex illumination conditions and irregularity of cracks from the training datasets. Third, the multiple nonlinear layers of the neural networks can represent the data better, which usually yields better performance.

Recently, significant progress has been made in crack detection methods using deep neural networks. Protopapadakis et al. [13] proposed an intelligent platform for the tunnel for tunnel assessment. On this platform, they developed a 3-layer convolutional neural network (CNN) for crack detection in tunnels. Cha et al. [22] trained an 8-layer CNN on 40,000 image patches with a resolution of 256 × 256 pixels. Each patch was classified as crack or non-crack. With the help of CNN, they could test the trained neural network on images with different resolutions. Zhang et al. [26] proposed a 5-layer deep neural network with more than 1 million parameters. With the help of convolutional layers and fully connected layers, the input and output of the neural network have the same size so that pixel-level prediction could be achieved. Ni et al. [27] developed a method integrating a feature extraction network and crack delineation network for concrete crack detection. In their method, the features were extracted from different levels of the first neural network and then were fed to the second neural network for detection. Dung and Anh [28] presented a fully convolutional neural network (FCN) for crack detection. Without using pooling layers and fully connected layers, the input and output could stay the same size that pixel-level prediction can be achieved. They applied their method to a concrete specimen during a cyclic loading test. Bang et al. [29] designed a deep neural network in an encoder-decoder style. The encoder of their neural network was based on residual neural network and the decoder was implemented using deconvolutional layers. The method was applied to road images collected from a black box camera on moving cars. Zhang et al. [30] introduced a method combining semantic segmentation and neighborhood fusion for crack detection. In their method, Sobel edge detectors were first applied to find localized patches. Then, SegNet was applied to the patches for crack detection. Yang et al. [31] introduced a feature pyramid and hierarchical boosting network (FPHBN) into a holistically-nested edge detection method for pixel-level pavement crack detection. The pyramid module can reserve the information from low-level features and hierarchical boosting architecture can help the deep neural network to pay more attention to hard examples. In recent years, there is an increasing number of studies about crack detection using deep learning-based methods. [4,32–37]

In this paper, a novel method based on densely connected deep neural network considering the connectivity of pixels in loss function is presented for pixel-level crack detection on road pavements. Feature fusion is conducted at multiple levels in the densely connected deep neural network. Densely connected neurons were first discussed in [38,39] in a random neural network. The corresponding learning strategy for this neural network was described in [40]. In these studies, each neuron was connected to at most 8 neighbors which help reserve the detailed information of the image. Then a recurrent random neural network was applied to extract precise morphometric information from magnetic resonance imaging of human brains [41]. Recently, the dense random neural network was extended to multiple layer architecture (RNN-MLA) in the deep learning field for classification tasks [42]. Although the current work presented in this paper also includes densely connected layers, the concept used in this paper is different than the previous work [42]. This study includes massive “skip connections” that connect layers that are farther away in the chain-like architecture which leads to better reuse of low-level features, while RNN-MLA includes numerous local interconnections called soma-to-soma interactions.

A new loss function considering the connectivity of pixels in cracks is developed to resolve the discontinuity issue existed in deconvolutional layers. It is well believed that the loss function would affect the performance of a deep learning algorithm significantly [43]. In most crack detection algorithms [22,28,29,44], the influence of loss function was not studied. Regular cross-entropy function was usually used to calculate the loss of crack and non-crack pixels equally. However, the training set of crack detection images are usually highly biased, which means there are many more non-crack pixels than crack pixels. Some studies used focal loss instead of regular cross-entropy loss to resolve this issue and obtained very good performance [45]. Another issue is that cracks on an image are highly local information which means if one pixel belongs to a crack the adjacent pixels are more likely to belong to the same crack. In the traditional cross-entropy loss function, all the pixels were measured independently, and no information about their connectivity was taken into consideration during the training. This would lead to the sparsely distributed output. The new loss function proposed in this paper is to resolve this issue. In the last step of the proposed method, a depth-first search (DFS) method is used to threshold crack pixels from the background.

The contribution of this paper includes the following: 1) a novel deep neural network architecture that densely connects multiple layers in a forward fashion is used for crack detection. 2) a novel loss function is proposed to take into consideration of connectivity of crack pixels; 3) this study designed and implemented a real-life experimental study to collect video extracted images using a camera mounted at the rear of a vehicle operating at traffic speed to mimic the behavior of a backup camera.

This paper is organized as follows. Section 2 describes the methodology and the proposed method to overcome existing issues. Section 3 presents two datasets and the corresponding analysis results. Conclusions are drawn in section 4.

2. Methodology

2.1. Overview

Fig. 1 presents the overall procedure of the proposed method. In step 1, the original image is divided into 256 × 256 patches for data augmentation. In step 2, the patches are fed into the deep neural network built mainly from the densely connected neural network and transposed convolution layers. The densely connected neural network has 201 layers, which is usually called DenseNet201 [46]. The feature fusion is conducted at multiple levels to generate a better prediction map. In step 3, the patches are integrated and binarized to reconstruct the crack binary mask in the third step. In step 4, a DFS algorithm is applied to find connected components in the binary mask. Based on the logic that cracks usually have a large number of connected pixels while noise often has much fewer pixels, we threshold the connected components with a number of pixels lower than a certain value to separate the cracks from the background.

2.2. Densely connected neural network with multiple level feature fusion

As presented in Fig. 1, the deep neural network consists of convolutional layers, max pooling layers, densely connected layers,

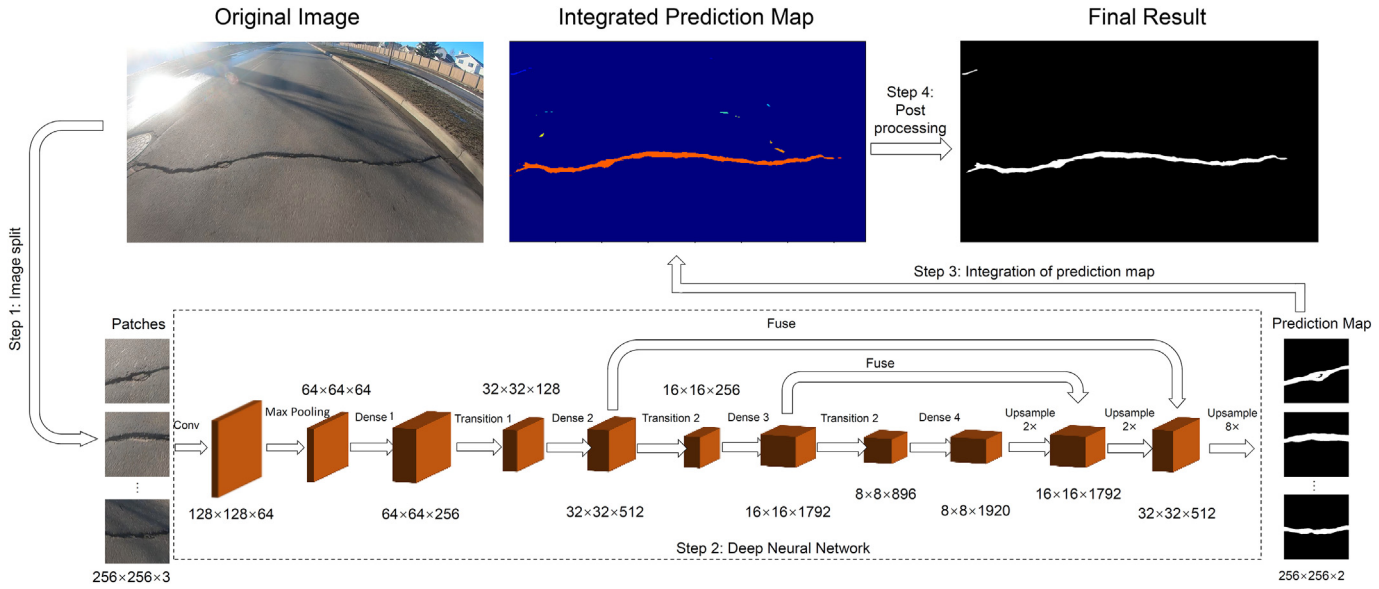

Fig. 1. Overview of the proposed method.

transition layers, and transposed convolution layers. The network follows an encoder-decoder schema where the width and height of the features become smaller as the mainstream of the neural work goes deeper. The feature fusion is done at multiple levels to decode the features and the original size of the image is recovered. First, the output of dense 4 layers ($8 \times 8 \times 1920$) is upsampled twice to features with $16 \times 16 \times 1792$. Then, the upsampled features are added to the output of a dense 3 layers. After that, the added features are upsampled again to the dimension of $32 \times 32 \times 512$. The features are further added with the output of the dense 2 layers. Eventually, the fused features are upsampled one last time to $256 \times 256 \times 2$ which is the original size of the image to achieve the pixel-level crack detection. The functions of different layers are summarized in the following sections.

2.2.1. Convolutional layer

The most important layer in the proposed deep neural network is called the convolutional layer, which was first proposed by LeCun et al. [47]. In this layer, the data is processed in the form of an operation called convolution [48]. The convolution operation is widely used in image processing [49], where the localized features are convolved with the kernel to generate new outputs. Unlike in traditional image processing techniques where the parameters in the kernel are predefined, the parameters in convolutional layers are determined during the training process. The equation for a convolutional layer is presented in Eq. (1).

$$O(i,j,k) = \sum_{m}\sum_{n}\sum_{l} I(i-m, j-n, l) w_k(m,n,l) + b_k \tag{1}$$

where I and O stand for the input and output features of the convolutional layer. The symbols l and k represent the lth and kth features in the input and output.

2.2.2. Max pooling layer

A max pooling layer modifies the input of the layer in a way that the value of features at a certain location is replaced with the maximum value of the nearby features [48]. The max pooling layer makes the features invariant to small translations of the input and can also improve the computational efficiency of the network. The equation of max pooling layer is

$$O(x,y,l) = \max_{p=-t}^{t}\max_{q=-t}^{t}(I(i-p, j-q, l)) \tag{2}$$

where i and j are indices of the input features, x and y are indices of the output and $2t + 1$ is the kernel size.

2.2.3. Dense block

It is believed that deeper neural networks can represent complex data better with more nonlinearity in previous studies [50]. However, traditional CNN is a chain-like architecture, and this configuration will make the neural networks harder to train when they become deeper due to the increasing number of parameters and gradient vanishing [51]. In order to resolve the training issues and gain a performance boost, Huang et al. [46] introduced a standard block called the dense block. Within such block, every layer is connected to every following layer in a feed-forward fashion except the mainstream chain-like structure. In this way, the gradient vanishing problem can be alleviated, and also the features can be reused so that the number of parameters can be significantly reduced.

Fig. 2 presents the details of dense block 1 in Fig. 1. The basic component of a dense block is a $1 \times 1 \times 128$ convolutional layer and a $3 \times 3 \times 32$ convolutional layer. The only difference among dense blocks 1 to 4 is the number of basic components. The dense block 1 has 6 such components, and dense blocks 2 to 4 have 12, 48 and 32 basic components, respectively. Taking dense block 1 as an example, the input has a dimension of $64 \times 64 \times 64$, and it is passed through the basic component to generate a feature of $64 \times 64 \times 32$. Then, feature 1 is concatenated with input to generate a feature of $64 \times 64 \times (64 + 32)$. Similarly, feature 2 is concatenated with feature 1 and input to generate a feature of $64 \times 64 \times (64 + 32 + 32)$. It is seen that the depth of feature is increased by 32 every time a basic component is passed through. The output of the dense block 1 has a dimension of $64 \times 64 \times (64 + 32 \times 6) = 64 \times 64 \times 256$. Similarly, the output of dense block 2 has a dimension of $32 \times 32 \times (128 + 32 \times 12) = 32 \times 32 \times 512$, dense block 3 has $16 \times 16 \times (256 + 32 \times 48) = 16 \times 16 \times 1792$, and dense block 4 has $8 \times 8 \times (896 + 32 \times 32) = 8 \times 8 \times 1920$.

2.2.4. Transition block

The dense block itself only changes the depth without touching the height and width of the features. The height and width of the features are changed using transition blocks. A transition block consists of a 1×1 convolutional layer with half the depth of the input and an average pooling layer with a size of 2×2 and a stride of 2. As presented in Fig. 3, height, width and depth are all divided by half by

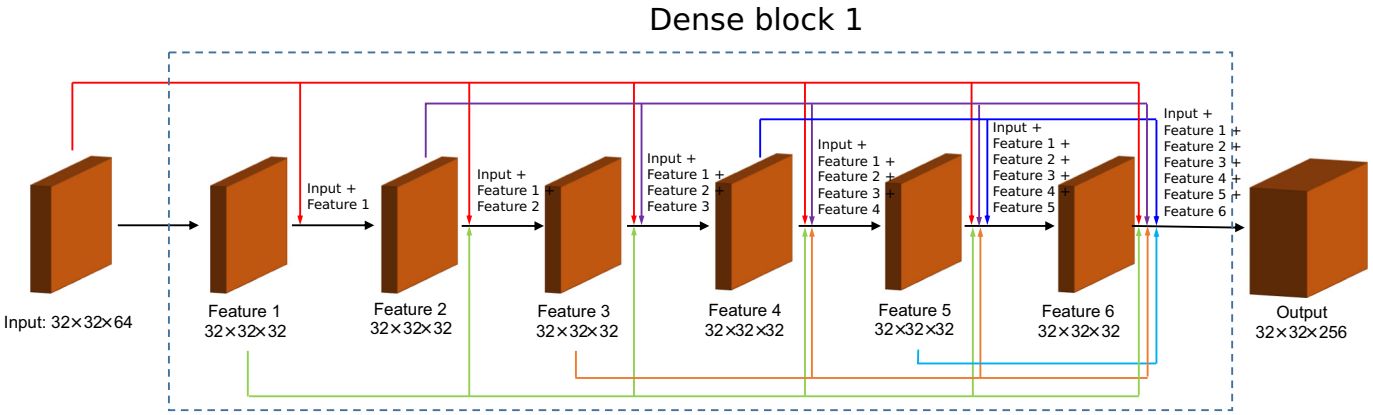

Fig. 2. Details of dense block 1.

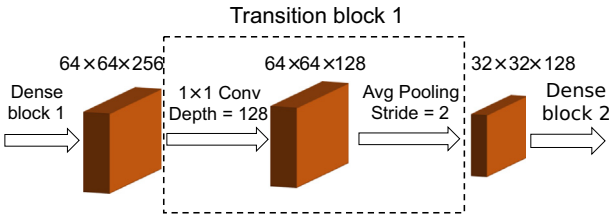

Fig. 3. Details of transition block 1.

passing through this block. The transition blocks are applied following every dense block to reduce the size of the features.

2.2.5. Transposed convolutional layer and feature fusion

In an encoder-decoder schema, the size of features reduces first and then increases. All the layers described above either decrease or keep the size of the features. The increase of the size is achieved by applying transposed convolution layers. Transposed convolution layer was used for image segmentation for the first time by Long et al. [52]. Similar but in a reverse way to a convolutional layer, the transposed layers conduct the upsampling in a sliding window form. The parameters in transposed convolution layers are trainable. As can be seen in Fig. 1, transposed layers are applied three times following the output of dense block 4. Every time the output is upsampled, it is added with the intermediate features from previous layers. In this way, the more detailed information from the early layers can be directly used for crack detection.

2.3. A new loss function

2.3.1. Problems with transposed convolution layers and traditional cross entropy loss function

As presented in the previous section, the transposed convolution layers conduct the upsampling to generate binary masks for cracks with the same width and height as the original patch. The parameters of the transposed convolution layers are determined during the training process. The pixels in the output is corresponding to a local area in the input image called the receptive field. However, each pixel is predicted independently even though the spatial relationship is reserved. In other words, there is no explicit restriction that forces one pixel to crack if all its neighboring pixels are cracks. The prediction will likely to be sparsely distributed. This issue was also observed in previous studies [29,44]. Morphological operations, i.e. a combination of dilation and erosion, are a possible option to this problem. However, the performance of this method is highly dependent on the chosen size of the operation. As can be seen in Fig. 4(c), the gaps among discontinued pixels cannot be completely filled if the size of morphological operations is too small. In Fig. 4(d), we can see some unnecessary area are filled when a large size is selected.

In addition, as shown in Eq. (3), regular cross entropy loss function treats the pixels at different locations indifferently.

$$\text{Loss} = \sum_{i,j \in image} [-y(i,j)\log\hat{y}(i,j) - (1-y(i,j))\log(1-\hat{y}(i,j))] \quad (3)$$

where $y(i,j)$ is the true label of a pixel at location i and j in the image, in which 1 represents crack and 0 represents non-crack. And $\hat{y}(i,j)$ is the label predicted by the deep neural network.

Fig. 5 presents sample crack annotation with 1 as crack and 0 as non-crack. It is seen that the pixel in the red box is surrounded by 8 crack pixels and the one in the green box is next to only 1 crack pixel. Obviously, the pixel that is surrounded by the crack pixels is more likely to be a crack and predicting it as non-crack would lead to output as shown in Fig. 4(b), i.e., pixels that belong to one crack segment are not connected. However, regular cross-entropy loss function treats the pixels in red and green boxes indifferently, and the wrong prediction in both pixels will result in the same loss. The problem with the above loss function is that it does not consider the relationship among annotations of neighboring pixels.

2.3.2. Loss function considering connectivity of pixels

In previous studies [28,29,31], the pixel-level crack annotations are treated as a binary mask, where the crack is 1 and non-crack is 0, and the cross-entropy loss function was used to calculate the correctness of the prediction of each pixel. The issues regarding this setting have been discussed in the last section. Although more advanced post-processing algorithms such as multiscale open-closing by reconstruction [53] can be applied to improve the performance, this paper considers the connectivity of pixels by designing a new loss function.

To resolve the problem mentioned earlier, we treat the pixel-level crack detection as a connectivity problem. A new loss function is presented to account for the connectivity of pixels while doing crack detection. First, we convert the binary mask annotation to connectivity maps. In Fig. 6, we can see each pixel P has 8 neighboring pixels. Therefore, 8 connectivity maps can be generated solely based on the binary mask information. As can be seen in Fig. 7, if a pixel is crack and is connected to its top-left neighbor (A1), the corresponding location of the A1 map is set to 1, and otherwise, it will be set to 0. For instance, in the binary mask of Fig. 7, the pixel at the second row and the second column is crack (1) and its top-left neighbor is also crack, so the element at the second row and second column of the A1 map is 1. In contrast, the top-left neighbor of the pixel at the second row and the fourth column is non-crack, so the corresponding element in the A1 map is 0.

The loss function is designed to optimize the neural network parameters so that all 8 connectivity maps are closer to the correct labels. The new loss function is a sum of the cross entropy function of all 8 connectivity maps.

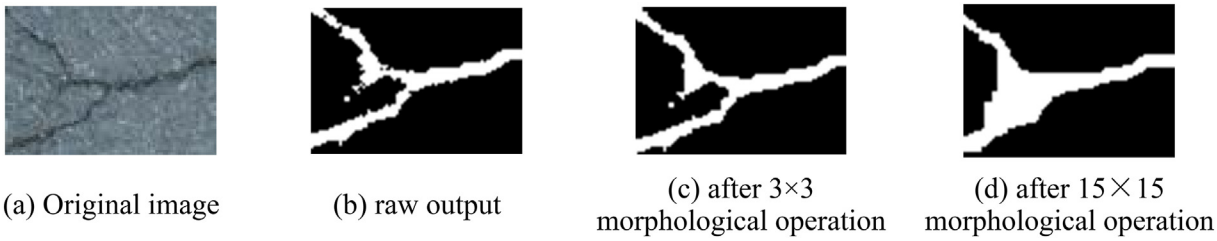

Fig. 4. Sample Result from the deep neural network with regular cross entropy loss function.

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Fig. 5. Sample crack annotation.

A1	A8	A7
A2	N/A	A6
A3	A4	A5

Fig. 6. Connectivity of Pixel P.

$$\text{New Loss} = \sum_{k=1}^{8} \sum_{i,j \in image} [-y_{A_k}(i,j) \log \hat{y}_{A_k}(i,j) - (1 - y_{A_k}(i,j)) \log(1 - \hat{y}_{A_k}(i,j))] \quad (4)$$

where $y_{A_k}(i,j)$ is the true label of a pixel at location i and j in the connectivity map A_k. And $\hat{y}_{A_k}(i,j)$ is the label predicted by the deep neural network.

In this way, predicting a pixel surrounded by 8 crack pixels would result in roughly 8 times penalty than predicting a pixel which is only connected to one crack pixel. Fig. 8 compares the regular loss function and our proposed loss function. The corresponding area of the results from two loss functions is marked with the same color. We can see that the results from regular loss function are more scattered while the results using our proposed function is more solid and more accurate.

For the inference stage, we first set a threshold, θ_{prob}, for the connectivity maps, i.e., any prediction larger than θ_{prob} would yield a 1 in the corresponding feature map. Then, we sum up the values of all 8 feature maps. The largest and smallest values in the summation would be 8 and 0, which means all the neighboring pixels are crack pixels and none of the neighboring pixels is crack. Then, we specify another threshold, θ_{conn}. The pixel is predicted as crack if the summation of all 8 feature maps is larger than θ_{conn}.

2.4. Post-processing

The raw prediction from the proposed neural network may still include some noise (see Fig. 1) even though the proposed loss function has improved this aspect significantly. A post-processing algorithm is proposed based on the fact that cracks often have a number of connected pixels while noise has much fewer. In this algorithm, the raw binary mask generated by the proposed neural network is first converted to a graph where every pixel predicted as the crack is a node and there is an edge between every pair of the neighboring nodes. Therefore, the problem is simplified as looking for connected components (CC) in the graph and threshold out the CC with a small number of pixels. A standard DFS algorithm is applied to find all the CCs due to its computational efficiency [54].

3. Experiments and analysis

Two datasets are employed to verify the proposed method, where the first one is a publicly available dataset called CFD [11] and the second one is a dataset created by our team named as EdmCrack1000 which includes 1000 images collected from a commercial-grade sports camera mounted on the rear of a moving car. In this paper, both datasets are trained on a desktop with Intel 8700 k CPU, 32GB memory and Nvidia Titan V GPU with 5120 CUDA cores. The details of the datasets and the analysis results are given in the following sections.

3.1. CFD dataset

The CFD dataset was first proposed by Shi et al. [11] to verify their

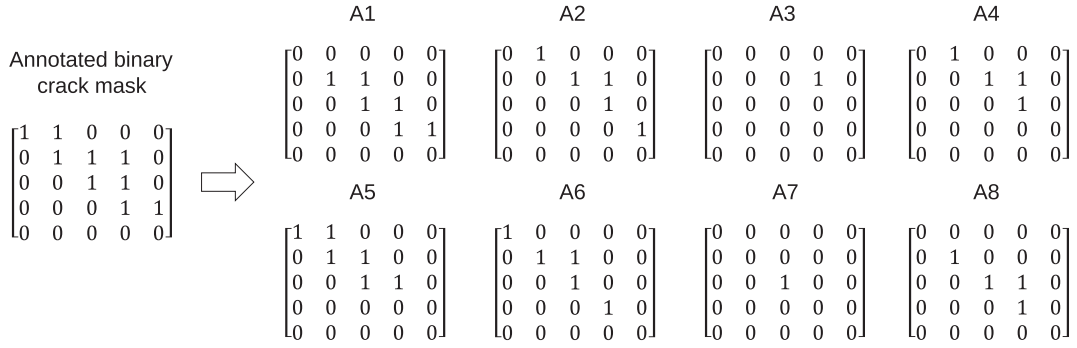

Fig. 7. Generating 8 connectivity maps.

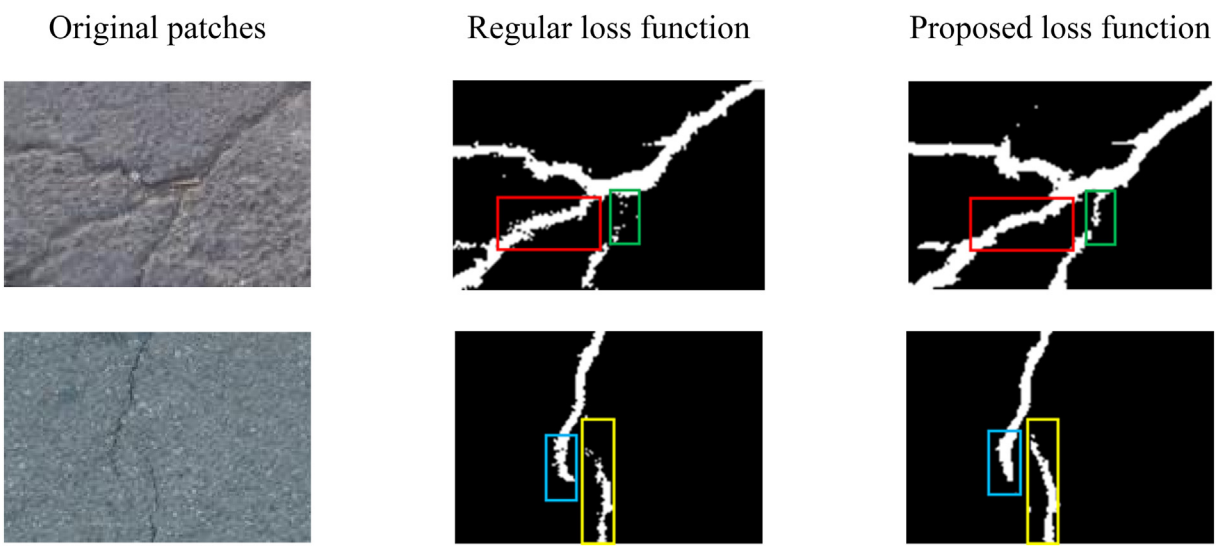

Fig. 8. Comparison between regular loss function and our proposed loss function.

crack forest method. In total, the dataset consists of 118 road surface images taken with a handheld iPhone 5 in Beijing. The exposure time is 1/134 s, the focus is 4 mm and the aperture is $f/2.4$. The distance between the smartphone and the road surface is about 1.5 m. The images have a resolution of 480 by 320 pixels. Most of the images intentionally focus on a small area which includes pavement texture and cracks. There are stains and shadows in the images, but they are manually selected to exclude irrelevant objects such as garbage, potholes or cars on the roads.

According to [11], the dataset is randomly split to 70 and 48 images for training and testing following the 60%/40% rule. Since the performance of the deep neural network would not be good if the amount of data is not enough, data augmentation techniques are applied in this paper. First, all the training images are flipped horizontally and vertically. Then, they are cut into 128 × 128 patches with a stride of 16. In this way, each image can generate 299 square patches. Eventually, there are 299 × 70 × 3 = 62,790 patches for the training process. Due to the limitation of the GPU memory, the batch size is chosen as 16 in this paper. We choose this as the first dataset to verify our proposed method because it is well annotated and many other studies used this dataset so that a direct comparison can be made with their methods.

3.2. Analysis and results for CFD dataset

The training results for the CFD dataset are presented in Fig. 9. Each epoch represents going through all the training data on a batch basis once, which is 62,790/16 = 3925 iterations for the CFD dataset. Three metrics, precision, recall, and F1 score, are reported to show the performance of the proposed method. The formulae to calculate these three metrics are presented in Eqs. (5), (6) and (7).

$$\text{precision} = \frac{TP}{TP + FP} \tag{5}$$

$$\text{recall} = \frac{TP}{TP + FN} \tag{6}$$

$$\text{F1 score} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \tag{7}$$

where *TP*, *FP*, and *FN* are the number of true positive, false positive and false negative for each image. Following the definitions of [11], if a pixel is identified as crack by the proposed method, and it is within 5-pixel distance to a pixel annotated as crack, this pixel is considered as true positive. In contrast, if a pixel is identified as crack but there is no true crack pixel within 5-pixel distance, it is considered as false positive. If a pixel is identified as non-crack, but is actually a crack pixel according to the annotation, then this pixel is a false negative.

In Fig. 9, the new loss function over training epochs is shown in the left plot, and three metrics described above are shown in the right plot. In the right plot, it is shown that as the number of training epochs increases, the recall first drops gradually, but the recall goes up dramatically. This is because initial weights identify all the pixels as crack, and thus there are no *FN* pixels. As the training goes on, precision, recall, and F1 score converge to 91.00%, 93.22%, and 91.99%, respectively. In Table 1, the proposed method is compared with some other methods, where Canny, CrackTree, FFA, CrackForest, MFCD,

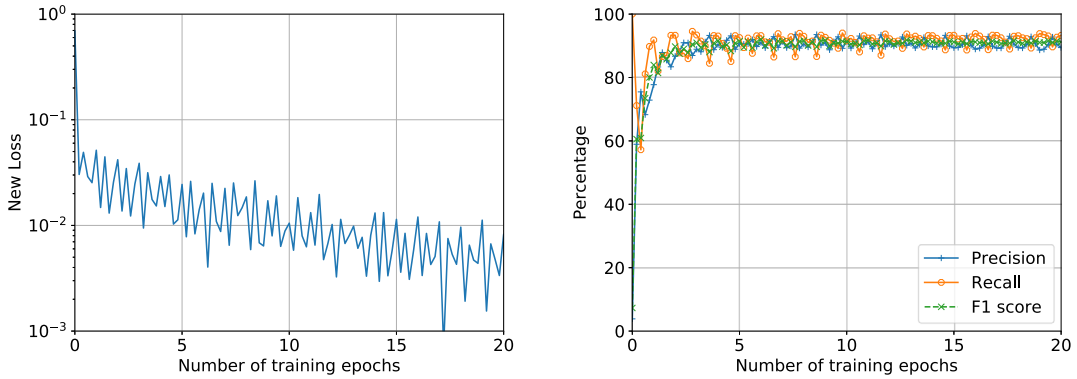

Fig. 9. Training results for CFD dataset: (left) new loss over training epochs; (right) precision, recall and F1 score over training epochs.

Table 1
Comparison of performance for different methods on CFD dataset.

Method	Precision	Recall	F1 score
Canny [11]	12.23%	22.15%	15.76%
CrackTree [11]	73.22%	76.45%	70.80%
FFA [55]	78.56%	68.43%	73.15%
CrackForest [11]	82.28%	89.44%	85.71%
ResNet152-FCN [29]	87.83%	88.19%	88.01%
MFCD [55]	89.90%	89.47%	88.04%
VGG19-FCN [44]	92.80%	85.49%	88.53%
CrackNet-V [56]	92.58%	86.03%	89.18%
UNet [45]	86.80%	77.97%	81.83%
FPHBN [31]	95.88%	87.97%	91.53%
Proposed method with regular loss function	92.02%	91.13%	91.58%
Proposed method with new loss function	91.00%	93.22%	91.99%

CrackNet-V are implemented by other researchers as cited in the table, and VGG19-FCN [28], ResNet152 [29], UNet [45] and FPHBN [31] are implemented by our team. CrackNet-V, VGG19-FCN, ResNet152-FCN, UNet, FPHBN and our proposed method are based on deep neural networks, while others are based on traditional image processing techniques. In all deep learning-based methods discussed in this section, the transposed convolution layers are applied as a way for feature fusion for pixel-level crack detection. It is seen that the proposed method can outperform other methods in terms of the F1 score. The new loss function can boost the F1 score from 91.58% to 91.99%. It should be noted that the UNet performs poorly on the CFD dataset mainly because it only supports 512 × 512 images as input, and the required dimensions are larger than the dimensions of the images in the CFD dataset. Therefore, the images have to be scaled to be fed into UNet, which might have caused the drop of performance.

Some sample images and results from the proposed method are presented in Fig. 10. We can see the proposed neural network with the novel loss function can identify the cracks very accurately and has smooth boundaries. The texture changes do not affect the identification process. The post-processing algorithm successfully removes the noise from the real cracks. The sample results from some other methods can be found in [11,55,56].

3.3. EdmCrack1000 dataset

To further verify the proposed method, a more challenging dataset collected by our team using a commercial-grade sports camera mounted on the rear of a moving vehicle is employed (see Fig. 11). Since current vehicles do not offer access to backup cameras, this experiment is set up to mimic the behavior of such cameras. There are mainly three reasons we choose to mount the camera at the rear of the vehicle: 1) the camera is installed outside of the car so that the influence of windows or windshield is avoided; 2) the camera is close to the ground and is not blocked by the hood so that the spatial resolution of the image is better. 3) the feasibility of using the backup camera in the vehicle for crack detection is studied so that no extra device is required if the access to the backup camera becomes more practical in the future. The camera used in this study is GoPro Hero 7 Black and the car is Honda Pilot 2017. Videos are continuously taken while the car is moving. The resolution of the videos is 1920 × 1080 pixels, and the frame rate is 240 fps. The images are extracted from more than 20 h of videos taken from the roads in Edmonton, Alberta, Canada. In this dataset, there are 1000 images annotated at the pixel-level by the first and third authors (QM and MRA) using software Sketchbook. The GoPro Hero 7 Black is 1 m

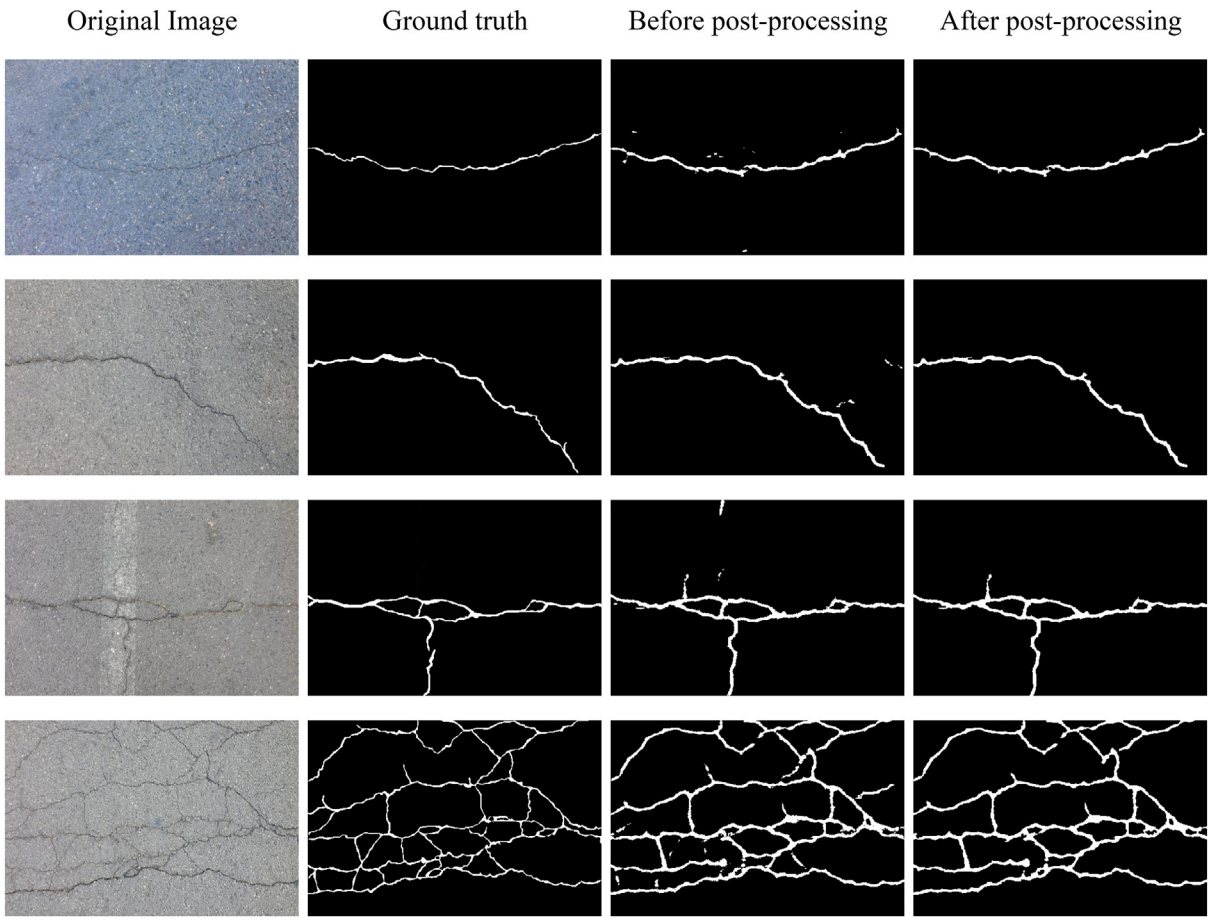

Fig. 10. Results from the proposed method for CFD dataset.

Fig. 11. The setup of the image collection system.

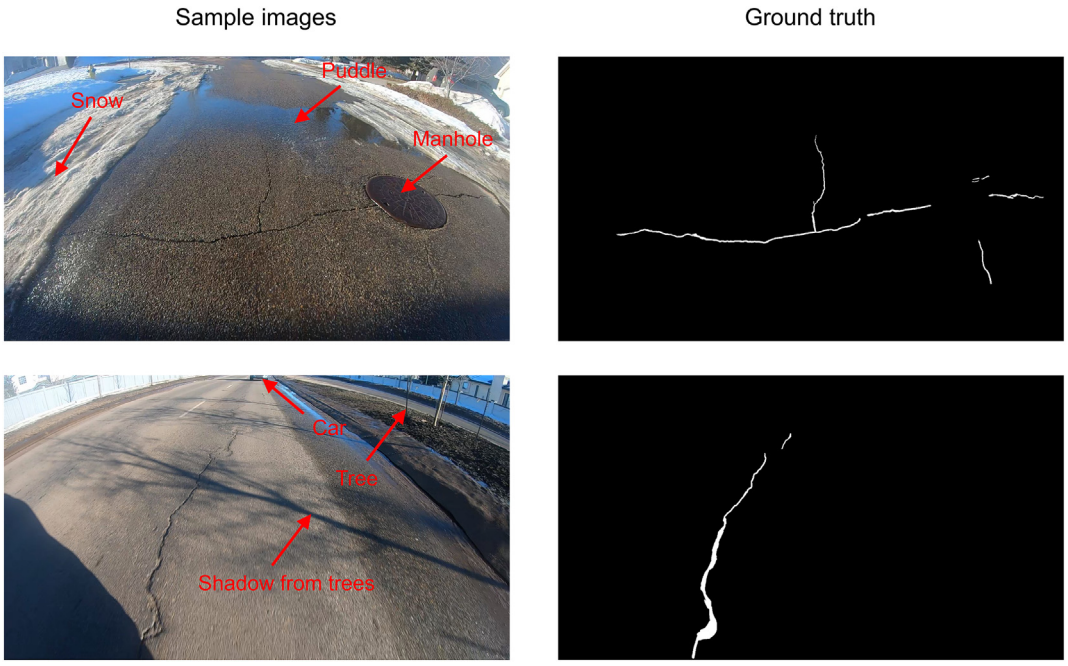

Fig. 12. Sample images from EdmCrack1000 along with the annotated ground truths.

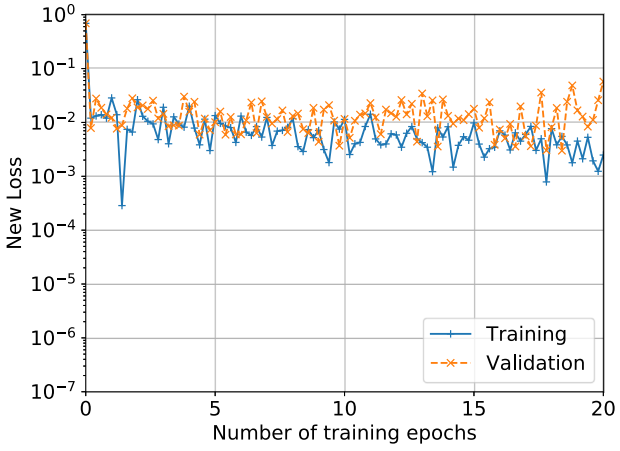

Fig. 13. New loss function during training.

Table 2
Comparison of performance for different methods on EdmCrack1000 dataset.

Methods	Precision	Recall	F1 score	Computational efficiency (*sec*/image)
Canny	1.56%	3.48%	2.92%	0.08
Sobel	2.42%	13.85%	3.89%	0.06
CrackIT	14.55%	7.36%	4.47%	5.57
VGG19-FCN	77.54%	61.88%	66.80%	3.05
ResNet152-FCN	78.37%	58.60%	64.05%	3.37
UNet	77.44%	67.83%	70.68%	4.83
FPHBN	61.63%	87.85%	71.41%	8.18
Proposed method with regular loss function	74.05%	72.31%	70.98%	3.46
Proposed method with new loss function	84.85%	70.59%	75.35%	3.96

from the ground and is facing downwards with an angle of 45°. There is no restriction about which objects should be included in the images to make the situation as real as possible. Because of the distance between the camera and the ground, the spatial resolution of the configuration can reach about 3 mm. Any cracks with a width larger than 3 mm will be distinguishable.

Fig. 12 shows two sample images from the EdmCrack1000 dataset. The cracks are annotated by binary masks, which will be converted to connectivity maps for our method. In addition to cracks, all other

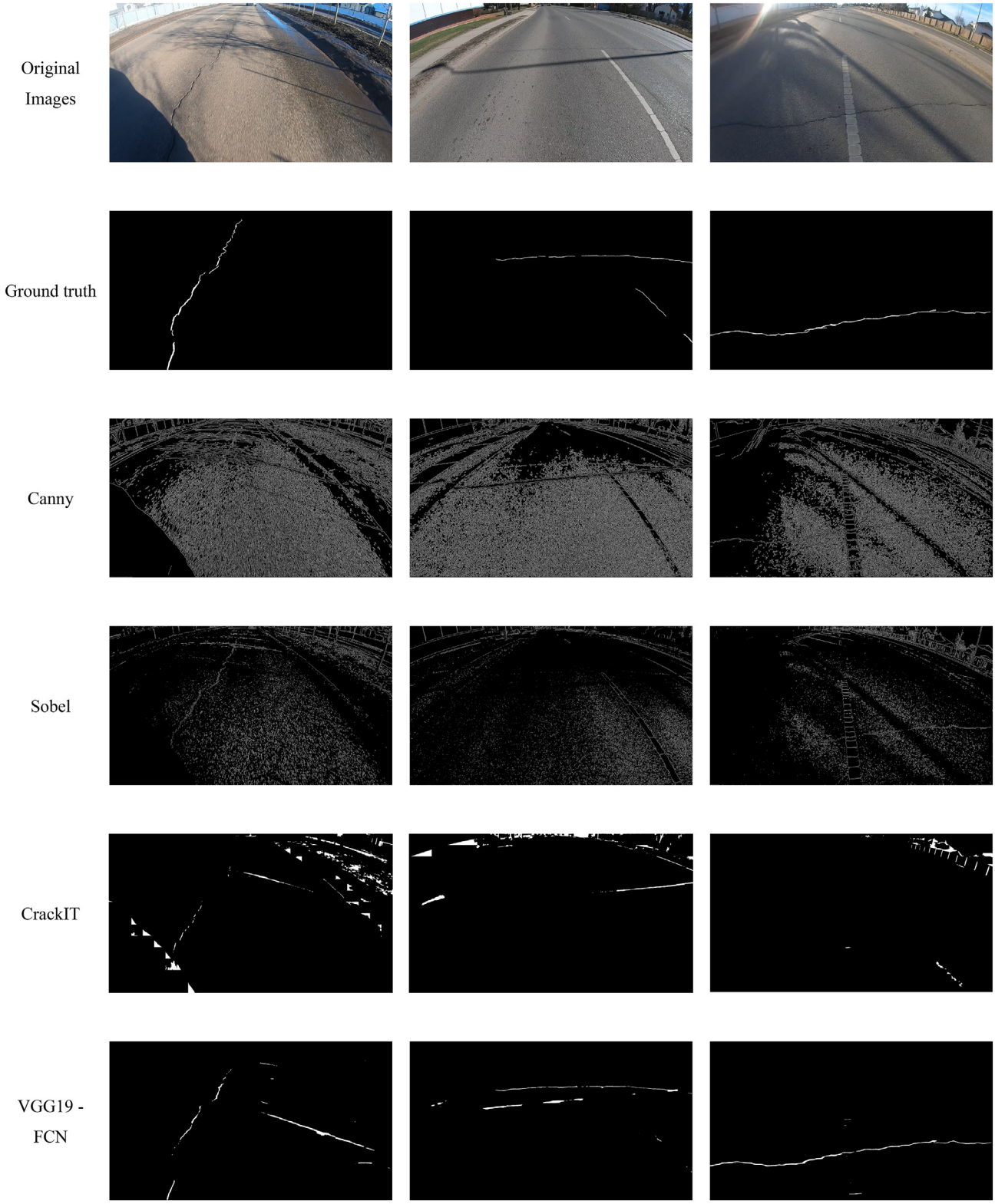

Fig. 14. Sample images and corresponding results.

objects that could appear on the roads are also included in the dataset. Different weather conditions and illumination conditions are also covered in the dataset. The goal of this dataset is to reflect the real road condition as much as possible.

For verification purposes, the dataset is split into training, validation and test sets, where the training set includes 700 images, and validation and test sets consist of 100 and 200 images, respectively. Similar to the CFD dataset, the images are split into 256 × 256 patches with a stride of 128 for training and testing. Therefore, we generate 112 patches for each image and there are in total 78,400 patches for training, 11,200 for validation and 22,400 for testing.

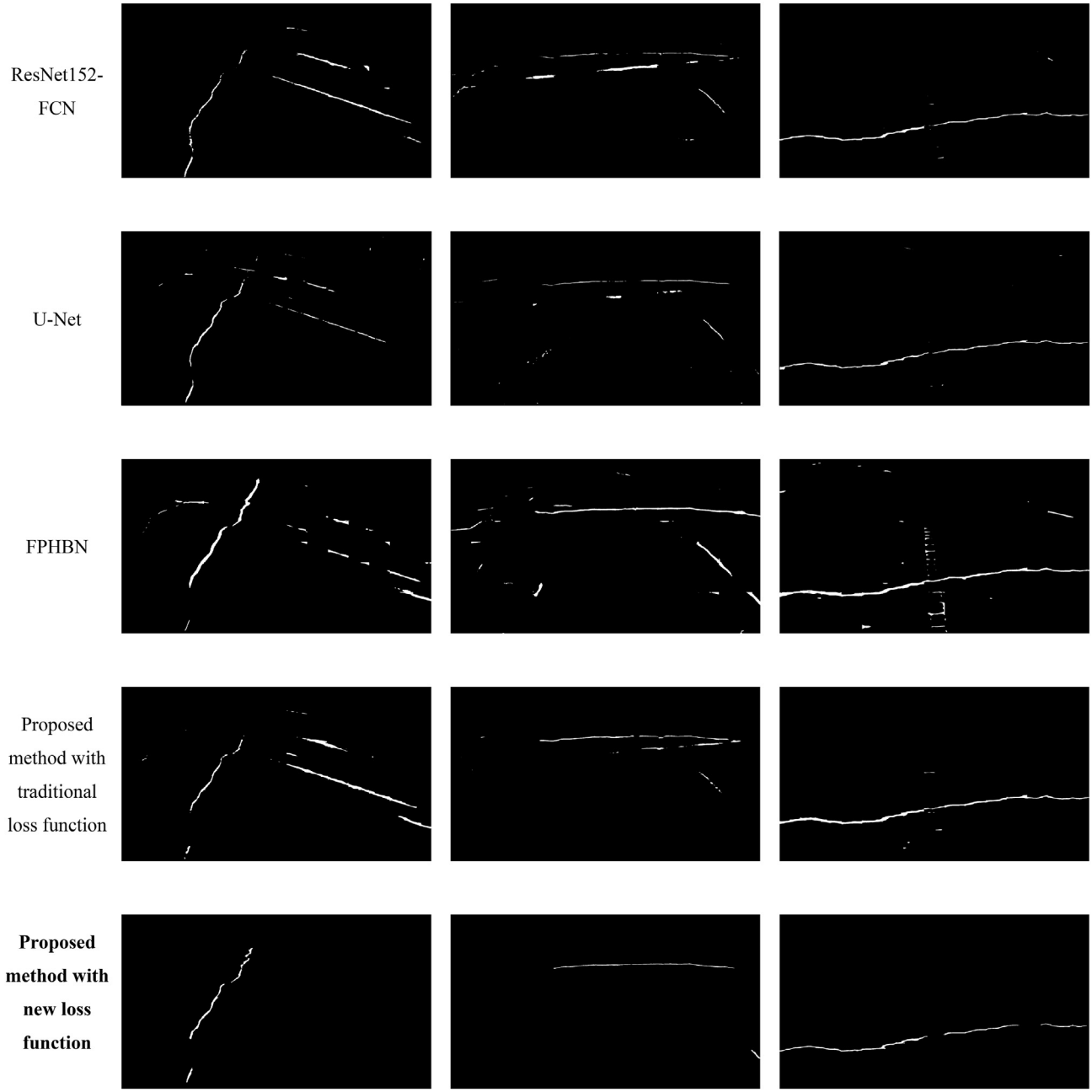

Fig. 14. (*continued*)

3.4. Analysis results for EdmCrack1000 dataset

The results for EdmCrack1000 dataset are shown in this section. Fig. 13 shows the new loss function values during the training process. The new losses on both training and validation sets are reported. They both reduce from 1 to 0.01 level and gradually converge as the training continues. The test set is applied at the end of 20 epochs.

The dataset is analyzed using our proposed method as well as 7 other methods among which Canny and Sobel are standard edge detection algorithms [49], CrackIT was proposed by [57,58], VGG19-FCN was proposed by [28], ResNet152-FCN was proposed by [29], UNet was proposed by [45] and FPHBN was proposed by [31]. Canny, Sobel and CrackIT methods are based on traditional image processing, while VGG19-FCN, ResNet152-FCN, UNet, and FPHBN are deep learning-based methods. Our proposed method has three advantages compared to the other four deep learning based-methods: first, a densely connected neural network is used to better reuse intermediate features; second, a novel loss function considering the connectivity of pixels is introduced; and third, a post-processing technique is introduced to improve the performance.

Table 2 presents the precision, recall and F1 score as defined in Section 3.2. They are 84.85%, 70.59%, and 75.35% respectively for our proposed method. It shows that our proposed method outperforms other methods in terms of all three metrics with a large margin. Canny, Sobel and CrackIT has very poor performance on this dataset, which makes sense because they were not designed to deal with such complex situations. VGG19-FCN [44], ResNet152-FCN [29], UNet [45] and FPHBN [31] give better performance than traditional image processing techniques but are still not comparable to our proposed method. In the above comparison, VGG19-FCN and ResNet152-FCN are replicated by ourselves and the parameters are proposed in original papers were used. UNet and FPHBN are run from open-source codes provided by the authors of the papers. The computational efficiency for each method is also presented in Table 2. It is seen that the proposed method takes

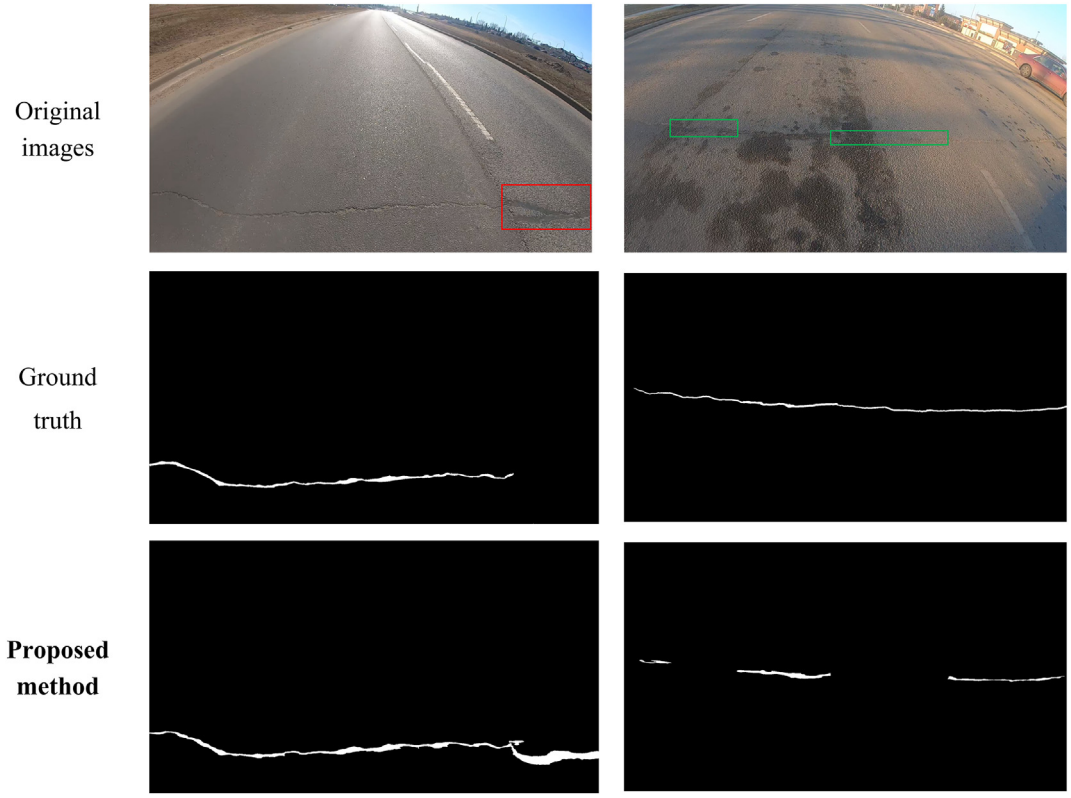

Fig. 15. Some wrongly identified images.

around 20–30% more time than VGG19-FCN and ResNet152-FCN but less time than UNet and FPHBN to process one image. The proposed new loss function can increase the F1 score from 70.95% to 75.35% for our method. It should be noted that previous studies have shown that random neural networks with multiple layer architecture (RNN-MLA) can outperform the CNN methods in classification tasks in terms of accuracy and time complexity [40]. However, the pixel-level crack detection task presented in this paper is different from the classification problem, and therefore RNN-MLA cannot be directly applied without significant changes. Although the proposed method is not quantitatively compared with a random neural network with multiple layer architecture (RNN-MLA), it is expected that integrating the idea of random neural network such as soma-to-soma interactions into our proposed method could further improve the accuracy and computational complexity. This will be studied in the future and is beyond the scope of this paper.

Some sample images from EdmCrack1000 and the results for all 7 other methods along with our proposed method with traditional loss function are shown in Fig. 14. It is clear that traditional methods, i.e., Canny, Sobel and CrackIT, cannot distinguish the cracks from the background very well in such a complex situation. VGG19-FCN, ResNet152-FCN, UNet, and FPHBN perform better on the given dataset, but some other objects like shadows from trees or edges of the roads are also wrongly identified as cracks. The results from our proposed method with new loss function are the best because it clearly identifies the cracks and is robust when shadow and illumination changes exist. It should be acknowledged that there are still some crack segments that are not properly identified in our proposed method.

Although the proposed method has achieved state-of-the-art performance, there are still some limitations to the current method. Fig. 15 presents some images that are not identified correctly. In the left image of Fig. 15, as can be seen in the red box, the sealed cracks are wrongly identified as real cracks with the proposed method. The reason could be that the training data do not include too many images with sealed cracks, and the method cannot recognize it properly because real and sealed cracks are indeed very similar in terms of aspect ratio and color intensities. In the right image, some parts of a long crack (in the green box) is not identified by the proposed method due to the disturbance of the stains. One possible solution for these issues is to collect a larger dataset with more critical cases.

4. Conclusions

The paper shows the feasibility of using a cost-effective device and a deep learning-based algorithm for pavement crack detection. In this paper, a novel deep neural network architecture with a new loss function considering the connectivity of pixels for automatic crack detection is proposed. In the proposed method, features are fused at multiple levels in a densely connected neural network to output pixel-level identification for cracks. The new loss function considering the connectivity of pixels is introduced to avoid scattered results and to make the boundaries smoother. The following conclusions are drawn from this study:

1. The proposed loss function tackled the issues regarding deep neural networks with transposed convolution layers, i.e., sparsely distributed identification, for pixel-level crack detection.
2. The proposed method outperforms other methods in terms of precision, recall and F1 score for two datasets.
3. It is feasible to use a deep learning-based method to detect cracks on the pavement in a complex environment.

Even though the proposed method achieves excellent performance on the given datasets, there are still challenges that are yet not resolved. The proposed method was tested on pavements only. Also, this method does not aim to distinguish from different types of cracks or road defects. Further studies should be conducted to address these issues. In addition, the performance of the algorithm, as well as many other

algorithms, is not good if a regular low-speed camera is used during driving because of the blurriness of images. Also, the low contrast due to low light conditions could degrade the performance. In the future, the authors will keep working on improving the performance of the proposed crack detection method for complex situations as well as reducing its computational complexity. Also, other practical issues encountered during the application of the proposed method will be studied.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] G. Félio, Informing the future: the Canadian infrastructure report card, http://canadianinfrastructure.ca/en/index.html, (October 18, 2019).

[2] R. Montero, J. Victores, E. Menendez, C. Balaguer, The Robot-spect Eu Project: Autonomous Robotic Tunnel Inspection, UC3M, (2015).

[3] K. Loupos, A.D. Doulamis, C. Stentoumis, E. Protopapadakis, K. Makantasis, N.D. Doulamis, A. Amditis, P. Chrobocinski, J. Victores, R. Montero, Autonomous robotic system for tunnel structural inspection and assessment, International Journal of Intelligent Robotics and Applications 2 (1) (2018) 43–66, https://doi.org/10.1007/s41315-017-0031-9.

[4] K. Makantasis, E. Protopapadakis, A. Doulamis, N. Doulamis, C. Loupos, Deep convolutional neural networks for efficient vision based tunnel inspection, 2015 IEEE International Conference on Intelligent Computer Communication and Processing (ICCP), IEEE, 2015, pp. 335–342, , https://doi.org/10.1109/ICCP.2015.7312681.

[5] A. Mohan, S. Poobal, Crack detection using image processing: a critical review and analysis, Alexandria Engineering Journal 57 (2) (2018) 787–798, https://doi.org/10.1016/j.aej.2017.01.020.

[6] J.-T. Kim, N. Stubbs, Crack detection in beam-type structures using frequency data, J. Sound Vib. 259 (1) (2003) 145–160, https://doi.org/10.1006/jsvi.2002.5132.

[7] S.-T. Quek, Q. Wang, L. Zhang, K.-K. Ang, Sensitivity analysis of crack detection in beams by wavelet technique, Int. J. Mech. Sci. 43 (12) (2001) 2899–2910, https://doi.org/10.1016/S0020-7403(01)00064-9.

[8] Q. Shan, R. Dewhurst, Surface-breaking fatigue crack detection using laser ultrasound, Appl. Phys. Lett. 62 (21) (1993) 2649–2651, https://doi.org/10.1063/1.109274.

[9] E. Glushkov, N. Glushkova, A. Ekhlakov, E. Shapar, An analytically based computer model for surface measurements in ultrasonic crack detection, Wave Motion 43 (6) (2006) 458–473, https://doi.org/10.1016/j.wavemoti.2006.03.002.

[10] G. Owolabi, A. Swamidas, R. Seshadri, Crack detection in beams using changes in frequencies and amplitudes of frequency response functions, J. Sound Vib. 265 (1) (2003) 1–22, https://doi.org/10.1016/S0022-460X(02)01264-6.

[11] Y. Shi, L. Cui, Z. Qi, F. Meng, Z. Chen, Automatic road crack detection using random structured forests, IEEE Trans. Intell. Transp. Syst. 17 (12) (2016) 3434–3445, https://doi.org/10.1109/TITS.2016.2552248.

[12] M.R. Jahanshahi, J.S. Kelly, S.F. Masri, G.S. Sukhatme, A survey and evaluation of promising approaches for automatic image-based defect detection of bridge structures, Struct. Infrastruct. Eng. 5 (6) (2009) 455–486, https://doi.org/10.1080/15732470801945930.

[13] E. Protopapadakis, C. Stentoumis, N. Doulamis, A. Doulamis, K. Loupos, K. Makantasis, G. Kopsiaftis, A. Amditis, Autonomous robotic inspection in tunnels, ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences 3 (5) (2016), https://doi.org/10.5194/isprs-annals-III-5-167-2016.

[14] Y. Turkan, J. Hong, S. Laflamme, N. Puri, Adaptive wavelet neural network for terrestrial laser scanner-based crack detection, Autom. Constr. 94 (2018) 191–202, https://doi.org/10.1016/j.autcon.2018.06.017.

[15] S.K. Sinha, P.W. Fieguth, Automated detection of cracks in buried concrete pipe images, Autom. Constr. 15 (1) (2006) 58–72, https://doi.org/10.1016/j.autcon.2005.02.006.

[16] Y. Fujita, Y. Hamamoto, A robust automatic crack detection method from noisy concrete surfaces, Mach. Vis. Appl. 22 (2) (2011) 245–254, https://doi.org/10.1007/s00138-009-0244-5.

[17] L. Abdel-Qader, O. Abudayyeh, M.E. Kelly, Analysis of edge-detection techniques for crack identification in bridges, J. Comput. Civ. Eng. 17 (4) (2003) 255–263, https://doi.org/10.1061/(ASCE)0887-3801(2003)17:4(255.

[18] S. Iyer, S.K. Sinha, A robust approach for automatic detection and segmentation of cracks in underground pipeline images, Image Vis. Comput. 23 (10) (2005) 921–933, https://doi.org/10.1016/j.imavis.2005.05.017.

[19] Y.-J. Cha, K. You, W. Choi, Vision-based detection of loosened bolts using the Hough transform and support vector machines, Autom. Constr. 71 (2016) 181–188, https://doi.org/10.1016/j.autcon.2016.06.008.

[20] D. Lecompte, J. Vantomme, H. Sol, Crack detection in a concrete beam using two different camera techniques, Struct. Health Monit. 5 (1) (2006) 59–68, https://doi.org/10.1177/1475921706057982.

[21] H.K. Jung, G. Park, Rapid and non-invasive surface crack detection for pressed-panel products based on online image processing, Struct. Health Monit. (2019) 1475921718811157, , https://doi.org/10.1177/1475921718811157.

[22] Y.J. Cha, W. Choi, O. Büyüköztürk, Deep learning-based crack damage detection using convolutional neural networks, J. Comput. Aided Civ. Infrastruct. Eng. 32 (5) (2017) 361–378, https://doi.org/10.1111/mice.12263.

[23] K. He, G. Gkioxari, P. Dollár, R. Girshick, Mask r-cnn, Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 2961–2969, , https://doi.org/10.1109/ICCV.2017.322.

[24] Y.J. Cha, W. Choi, G. Suh, S. Mahmoudkhani, O. Büyüköztürk, Autonomous structural visual inspection using region-based deep learning for detecting multiple damage types, J. Comput. Aided Civ. Infrastruct. Eng. 33 (9) (2018) 731–747, https://doi.org/10.1111/mice.12334.

[25] F.-C. Chen, M.R. Jahanshahi, NB-CNN: deep learning-based crack detection using convolutional neural network and naive Bayes data fusion, IEEE Trans. Ind. Electron. 65 (5) (2018) 4392–4400, https://doi.org/10.1109/TIE.2017.2764844.

[26] A. Zhang, K.C. Wang, B. Li, E. Yang, X. Dai, Y. Peng, Y. Fei, Y. Liu, J.Q. Li, C. Chen, Automated pixel-level pavement crack detection on 3D asphalt surfaces using a deep-learning network, J. Comput. Aided Civ. Infrastruct. Eng. 32 (10) (2017) 805–819, https://doi.org/10.1111/mice.12297.

[27] F. Ni, J. Zhang, Z. Chen, Pixel-level crack delineation in images with convolutional feature fusion, Struct. Control. Health Monit. (2019) e2286, https://doi.org/10.1002/stc.2286.

[28] C.V. Dung, L.D. Anh, Autonomous concrete crack detection using deep fully convolutional neural network, Autom. Constr. 99 (2019) 52–58, https://doi.org/10.1016/j.autcon.2018.11.028.

[29] S. Bang, S. Park, H. Kim, H. Kim, Encoder–decoder network for pixel-level road crack detection in black-box images, J. Comput. Aided Civ. Infrastruct. Eng. (2019), https://doi.org/10.1111/mice.12440.

[30] X. Zhang, D. Rajan, B. Story, Concrete crack detection using context-aware deep semantic segmentation network, J. Comput. Aided Civ. Infrastruct. Eng. (2019), https://doi.org/10.1111/mice.12477.

[31] F. Yang, L. Zhang, S. Yu, D. Prokhorov, X. Mei, H. Ling, Feature pyramid and hierarchical boosting network for pavement crack detection, IEEE Trans. Intell. Transp. Syst. (2019), https://doi.org/10.1109/TITS.2019.2910595.

[32] C. Feng, M.-Y. Liu, C.-C. Kao, T.-Y. Lee, Deep active learning for civil infrastructure defect detection and classification, J. Comput. Civ. Eng. 2017 (2017) 298–306, https://doi.org/10.1061/9780784480823.036.

[33] L. Zhang, F. Yang, Y.D. Zhang, Y.J. Zhu, Road crack detection using deep convolutional neural network, IEEE International Conference on Image Processing (ICIP), IEEE, 2016, pp. 3708–3712.

[34] A. Doulamis, N. Doulamis, E. Protopapadakis, A. Voulodimos, Combined convolutional neural networks and fuzzy spectral clustering for real time crack detection in tunnels, 2018 25th IEEE International Conference on Image Processing (ICIP), IEEE, 2018, pp. 4153–4157, , https://doi.org/10.1109/ICIP.2018.8451758.

[35] E. Protopapadakis, N. Doulamis, Image based approaches for tunnels' defects recognition via robotic inspectors, International Symposium on Visual Computing, Springer, 2015, pp. 706–716, , https://doi.org/10.1007/978-3-319-27857-5_63.

[36] H. Kim, E. Ahn, M. Shin, S.-H. Sim, Crack and noncrack classification from concrete surface images using machine learning, Struct. Health Monit. 18 (3) (2019) 725–738 doi:10.1177%2F1475921718768747.

[37] K. Jang, N. Kim, Y.-K. An, Deep learning–based autonomous concrete crack evaluation through hybrid image scanning, Struct. Health Monit. 18 (5-6) (2018) 1722–1737.

[38] V. Atalay, E. Gelenbe, N. Yalabik, The random neural network model for texture generation, Int. J. Pattern Recognit. Artif. Intell. 6 (1) (1992) 131–141, https://doi.org/10.1142/S0218001492000072.

[39] V. Atalay, E. Gelenbe, Parallel algorithm for colour texture generation using the random neural network model, Int. J. Pattern Recognit. Artif. Intell. 6 (02n03) (1992) 437–446, https://doi.org/10.1142/S0218001492000266.

[40] E. Gelenbe, Learning in the recurrent random neural network, Neural Comput. 5 (1) (1993) 154–164, https://doi.org/10.1162/neco.1993.5.1.154.

[41] E. Gelenbe, Y. Feng, K.R.R. Krishnan, Neural network methods for volumetric magnetic resonance imaging of the human brain, Proc. IEEE 84 (10) (1996) 1488–1496, https://doi.org/10.1109/5.537113.

[42] E. Gelenbe, Y. Yin, Deep learning with dense random neural networks, International Conference on Man–Machine Interactions, Springer, 2017, pp. 3–18, , https://doi.org/10.1016/j.procs.2018.07.183.

[43] T.-Y. Lin, P. Goyal, R. Girshick, K. He, P. Dollár, Focal loss for dense object detection, Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 2980–2988, , https://doi.org/10.1109/ICCV.2017.324.

[44] X. Yang, H. Li, Y. Yu, X. Luo, T. Huang, X. Yang, Automatic pixel-level crack detection and measurement using fully convolutional network, J. Comput. Aided Civ. Infrastruct. Eng. 33 (12) (2018) 1090–1109, https://doi.org/10.1111/mice.12412.

[45] Z. Liu, Y. Cao, Y. Wang, W. Wang, Computer vision-based concrete crack detection using U-net fully convolutional networks, Autom. Constr. 104 (2019) 129–139, https://doi.org/10.1016/j.autcon.2019.04.005.

[46] G. Huang, Z. Liu, L. Van Der Maaten, K.Q. Weinberger, Densely connected convolutional networks, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 4700–4708, , https://doi.org/10.1109/CVPR.2017.243.

[47] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proc. IEEE 86 (11) (1998) 2278–2324, https://doi.org/10.1109/5.726791.

[48] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, 2016.

[49] R.C. Gonzalez, P. Wintz, Digital Image Processing, Addison Wesley, 1987.

[50] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1–9, , https://doi.org/10.1109/CVPR.2015.7298594.

[51] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778, , https://doi.org/10.1109/CVPR.2016.90.

[52] J. Long, E. Shelhamer, T. Darrell, Fully convolutional networks for semantic segmentation, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3431–3440, , https://doi.org/10.1109/CVPR.2015.7298965.

[53] A. Doulamis, N. Doulamis, P. Maragos, Generalized multiscale connected operators with applications to granulometric image analysis, Proceedings 2001 International Conference on Image Processing (Cat. No. 01CH37205), 3 IEEE, 2001, pp. 684–687, , https://doi.org/10.1109/ICIP.2001.958211.

[54] T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein, Introduction to Algorithms, MIT Press, 2009.

[55] H. Li, D. Song, Y. Liu, B. Li, Automatic pavement crack detection by multi-scale image fusion, IEEE Trans. Intell. Transp. Syst. (99) (2018) 1–12, https://doi.org/10.1109/TITS.2018.2856928.

[56] Y. Fei, K.C. Wang, A. Zhang, C. Chen, J.Q. Li, Y. Liu, G. Yang, B. Li, Pixel-level cracking detection on 3D asphalt pavement images through deep-learning-based CrackNet-V, IEEE Trans. Intell. Transp. Syst. (2019), https://doi.org/10.1109/TITS.2019.2891167.

[57] H. Oliveira, P.L. Correia, CrackIT — an image processing toolbox for crack detection and characterization, 2014 IEEE International Conference on Image Processing (ICIP), 2014, pp. 798–802, , https://doi.org/10.1109/ICIP.2014.7025160.

[58] H. Oliveira, P.L. Correia, Automatic road crack detection and characterization, IEEE Trans. Intell. Transp. Syst. 14 (1) (2012) 155–168, https://doi.org/10.1109/TITS.2012.2208630.