基于深度学习的桥梁裂缝检测算法研究

李良福[1] 马卫飞[1] 李 丽[1] 陆 铖[1]

摘 要 传统的图像处理算法不能很好的对桥梁裂缝进行检测, 而经典的深度学习模型直接用于桥梁裂缝的检测, 效果不甚理想. 针对这些问题, 本文提出了一种基于深度学习的桥梁裂缝检测算法. 首先, 利用滑动窗口算法将桥梁裂缝图像切分为较小的桥梁裂缝面元图像和桥梁背景面元图像, 并根据对面元图像的分析, 提出一种基于卷积神经网络 (CNN) 的 DBCC (Deep-bridge-crack-classify) 分类模型, 用于桥梁背景面元和桥梁裂缝面元的识别. 然后, 基于 DBCC 分类模型结合改进的窗口滑动算法对桥梁裂缝进行检测. 最后, 采用图像金字塔和 ROI 区域结合的搜索策略对算法进行加速. 实验结果表明: 与传统算法相比, 本文算法具有更好的识别效果和更强的泛化能力.

关键词 裂缝检测, 深度学习, 卷积神经网络, 窗口滑动算法

引用格式 李良福, 马卫飞, 李丽, 陆铖. 基于深度学习的桥梁裂缝检测算法研究. 自动化学报, 2018, xx(x): xxx-xxx

DOI 10.16383/j.aas.2018.c170052

Research on Detection Algorithm for Bridge Cracks based on Deep Learning

Li Liang-Fu[1] Ma Wei-Fei[1] Li Li[1] Lu Cheng[1]

Abstract The traditional image processing algorithms failed to detect the bridge cracks and the effect was not ideal if the classical deep learning models were used to detect the bridge cracks directly. In order to solve these problems, an algorithm for the detection of bridge cracks based on deep learning was proposed. Firstly, the bridge images with cracks were divided into smaller bridge crack patches and bridge background patches by using the window sliding algorithm. According to the analysis of the patches, a classification model based on convolutional neural network, called DBCC, was proposed and the model was used to identify the bridge background patches and bridge crack patches. Secondly, the DBCC classification model combined with improved window sliding algorithm was used for the detection of bridge cracks. Finally, the algorithm was accelerated by using a search strategy of combining image pyramid and ROI. The experimental results show that the algorithm has better recognition effect and stronger generalization ability compared with the traditional algorithm.

Key words Crack detection, deep learning, convolutional neural network, window sliding algorithm

Citation Li Liang-Fu, Ma Wei-Fei, Li Li, Lu Cheng. Research on detection algorithm for bridge cracks based on deep learning. Acta Automatica Sinica, 2018, xx(x): xxx-xxx

近年来, 我国的公路桥梁建设取得了空前发展, 特别是在《国家公路网规划 2013 年 -2030 年》和《十三五规划纲要》的推动下, 中国的桥梁总数已稳居世界第一[1]. 随着桥梁的建成通车, 桥梁的维护和管理成为保障桥梁安全运营的关键. 桥梁的质量安全关系国计民生, 关系千家万户. 桥梁垮塌的原因往往都是由于没有进行科学及时的桥梁病害检测, 因此, 必须选择科学合理的方法对桥梁病害进行检查, 定期的对其健康状况进行评估[2]. 裂缝作为最主要的桥梁病害之一, 严重影响着桥梁的安全运营, 更为严重的会发生桥毁人亡的事故. 因此, 对桥梁裂缝进行有效的检测识别至关重要. 随着计算机技术的高速发展, 特别是图像处理、模式识别与计算机视觉技术的发展, 基于图像的无损检测技术已经成为了国内外桥梁缺陷检测的研究热点. 近年来, 为了从影像中准确、快速、高效的提取裂缝, 国内外学者对此进行了广泛而深入的研究, 并且取得了一些研究成果. Oh 等人针对路面裂缝的检测和提取, 提出了一种迭代阈值分割的方法[3]; Li 等人为了从影像中准确的提取裂缝, 提出了一种基于相邻差分直方图的裂缝分割算法[4]; 这类基于阈值分割的裂缝识别方法虽然简单易用, 但是并没有考虑影像表面环境的变化以及光照、噪声、纹理对于裂缝识别的影响, 因此, 很难取得稳定的效果. 针对此问题, Landstrom 等人结

收稿日期 2017-01-20 录用日期 2018-02-07
Manuscript received January, 2017; accepted February 7, 2018
国家自然科学基金 (61573232, 61401263) 资助, 国家自然科学基金 (61401263) 资助, 中央高校基本科研业务费专项资金 (GK201703056) 资助.
National Natural Science Foundation of China (61573232), National Natural Science Foundation of China (61401263), the fundamental research funds for the central universities (GK201703056)
本文责任编委 王 亮
Recommended by Associate Editor WANG Liang
1. 陕西师范大学计算机学院 西安 710119
1.School of Computer Science, Shaanxi Normal University, Xi,an, 710119

合形态学处理和逻辑回归算法对裂缝进行检测, 利用统计学分类方法滤除噪声, 提高检测精度[5]; Varadharajan 等人基于对路面纹理、颜色和局部信息的分析, 对城市路面的裂缝进行了研究[6]; Jahanshahi 等人通过融合裂缝的深度信息, 提出了一种新的裂缝检测和评估方法[7]; Amhaz 和 Zou 等人针对路面裂缝的对比度低、连续性差等特点, 提出了一种基于最小代价路径搜索的路面裂缝检测算法[8−9]; Oliveira 等人构造了两个分类器, 然后, 用这两个分类器分别对路面裂缝进行检测和分类[10]; Nguyen 等人在综合考虑裂缝的亮度和连通性之后, 提出了一种用于道路裂缝检测的算法, 简称 FFA 算法[11]. 除了上述这些主流的裂缝检测算法之外, 还有很多其他的裂缝检测算法[12−19]. 但是, 桥梁裂缝图像不同于传统的路面裂缝、岩石裂缝图像, 它具有很多复杂的特性, 比如背景纹理多样复杂、噪声种类繁多、分布无规律等. 因此, 传统的裂缝检测算法不能很好的对桥梁裂缝进行检测.

深度学习, 特别是深度学习中的卷积神经网络 (CNN) 最近在图像识别、视频识别、语音识别中取得了巨大成功[20−21]. 但是, 经典的深度学习模型主要是针对大尺寸、整体目标的分类模型或者检测模型, 比如 AlexNet[22]、GoogLeNet[23−24]、Faster R-CNN 系列模型[25−26] 等, 例如采用 Faster R-CNN 模型检测到的目标如图 1 (a) 和图 1 (b) 所示. 如果将这些模型直接用于如图 1 (c) 所示的桥梁裂缝, 由于它是具有拓扑结构的线性目标, 效果将会十分的不理想, 只能得到一个目标区域块, 而得不到裂缝的具体位置, 如图 1 (d) 所示. 因此, 本文提出了一种基于深度学习的桥梁裂缝检测算法. 该算法首先利用滑动窗口算法将桥梁裂缝图像切分为 16×16 像素大小的切片, 并将所有的切片分为桥梁裂缝面元和桥梁背景面元. 然后, 根据对这些面元图像的分析, 提出一种基于卷积神经网络 (CNN) 的 DBCC (Deep-bridge-crack-classify) 分类模型, 用于桥梁背景面元和桥梁裂缝面元的识别. 最后, 结合改进的窗口滑动算法在整幅桥梁裂缝图像中对桥梁裂缝进行检测. 同时, 为了满足桥梁裂缝检测算法实时处理的要求, 采用图像金字塔和 ROI 区域相结合的搜索策略对算法进行加速. 大量的实验数据表明, 与传统算法相比, 本文算法具有更好的识别率和更强的泛化能力.

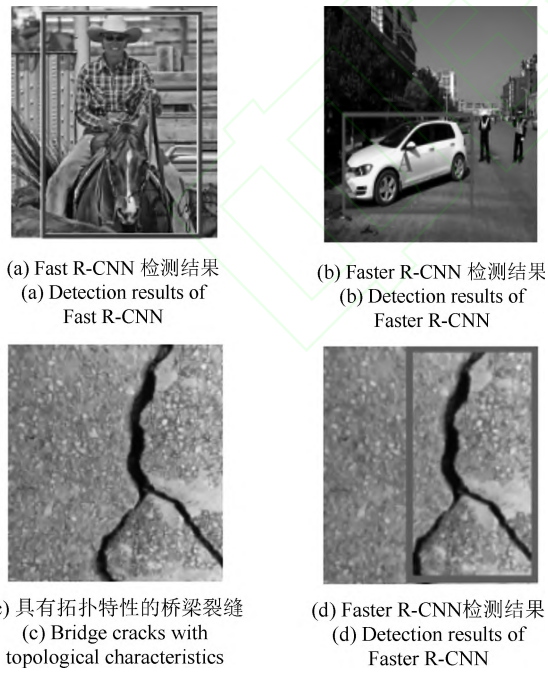


图 1 经典深度学习模型和桥梁裂缝特点示意图

Fig. 1 Schematic diagram of classical depth learning model and bridge crack characteristics

1 基于人工扩增的数据集预处理方法

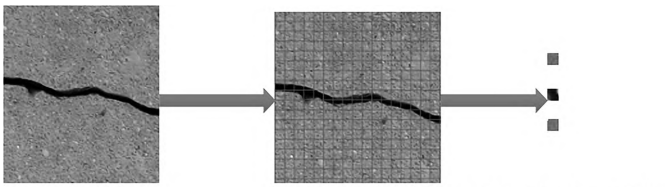
使用深度学习中的卷积神经网络 (CNN) 进行桥梁裂缝检测, 需要大量的、带类别标签的桥梁裂缝图像作为训练集、验证集和测试集. 但是, 到目前为止, 全球还没有公开的、带类别标签的、用于深度学习的桥梁裂缝图像数据集. 如果直接用人工的方式去采集大量的桥梁裂缝图像, 这将是个非常严峻的问题. 本文在对采集来的 2 000 张桥梁裂缝图像研究的基础之上, 提出一种基于滑动窗口算法, 专门用于桥梁裂缝图像数据集的人工扩增方法.

该人工扩增方法首先将采集来的桥梁裂缝图像归一化为 1 024×1 024 分辨率的桥梁裂缝图像. 然后, 使用 $W \times H$ 固定大小的窗口在桥梁裂缝图像上不重叠的进行滑动. 同时, 把窗口覆盖下的桥梁裂缝图像的小切片作为一个 ROI 感兴趣区域. 其中, 把不包含桥梁裂缝小切片的图像称为桥梁背景面元, 把包含桥梁裂缝的小切片称为桥梁裂缝面元, 具体过程如式 (1) 所示. 其中, W 和 H 为滑动窗口的宽和高, 并且取 $W = H = 16$ 像素; 坐标 $(imgRoiL_x, imgRoiL_y)$ 为 ROI 区域左上角的角点坐标, 坐标 $(imgRoiR_x, imgRoiR_y)$ 为 ROI 区域的右下角角点坐标, i 和 j 的取值范围分别为 $(0, 1, 2, \cdots srcImg_w/W)$ 和 $(0, 1, 2, \cdots srcImg_h/H)$. $srcImg_w$ 和 $srcImg_h$ 分别为被窗口滑动的桥梁裂缝图像的宽和高, 令 $srcImg_w = srcImg_h = 1\,024$ 像素. 最终, 整个数据集人工扩增的过程如图 2 所示.

通过将采集来的 2 000 张桥梁裂缝图像随机的划分为两个集合, 每个集合包含 1 000 张桥梁裂缝图像, 并且将这两个集合分别命名为 A 集合和 B 集

合. 对 A 集合中的 1 000 张桥梁裂缝图像使用上述

$$\begin{cases} imgRoiL_x = i \times W \\ imgRoiL_y = j \times H \\ imgRoiR_x = i \times W + W \\ imgRoiR_y = j \times H + H \end{cases} \tag{1}$$



(a) 原始图像 (a) Original image　(b) 切分示意图 (b) Cut-off schematic diagram　(c) 切分成的小 ROI 图像 (c) Segmented small ROI images

图 2　桥梁裂缝面元数据集人工扩增方式示意图

Fig. 2　Schematic diagram of manual expansion of bridge crack surface metadata set

的数据集人工扩增方式进行数据集扩增; 然后, 从这个巨大的、扩增之后的数据集合中手动的挑选出 7 000 张桥梁裂缝面元图像构成桥梁裂缝面元数据集合, 手动的挑选出 48 000 张桥梁背景面元图像构成桥梁背景面元图像的数据集合. 最后, 将桥梁裂缝面元集合和桥梁背景面元集合拆分为训练集和验证集. 其中, 训练集包含 6 000 张桥梁裂缝面元图像, 44 000 张桥梁背景面元图像; 验证集包含 1 000 张桥梁桥梁裂缝面元图像和 4 000 张桥梁背景面元图像. 在训练集和验证集中, 桥梁背景面元图像所占的比例大于桥梁裂缝面元图像所占的比例, 之所以这样划分, 是因为在一幅桥梁裂缝图像中, 桥梁裂缝区域所占整幅图像的比例小于桥梁背景区域所占整幅图像的比例. 训练集和验证集构建成功之后, 分别对训练集和验证集中的桥梁裂缝面元图像和桥梁背景面元图像构建相应的类别标签. 剩下 B 集合中的 1 000 张桥梁裂缝图像构成最终算法的测试集合. 此外, 本文为了推动桥梁裂缝自动检测与识别技术的进一步发展, 对本文使用的所有数据集进行了公开.

2　基于CNN 深度学习的 DBCC 分类模型及构建方法

2.1　DBCC 模型提出的原因分析

由图 3 可知, 桥梁裂缝图像具有背景纹理复杂多样、噪声种类繁多、分布无规律的特点, 因此, 主流的裂缝检测方法对于桥梁裂缝的检测, 其效果不甚理想. 由于深度学习中的卷积神经网络具有非常强大的特征提取能力和目标识别能力. 因此, 本文提出了一种基于 CNN 深度学习的 BDCC 分类模型, 用于识别桥梁裂缝面元和桥梁背景面元.

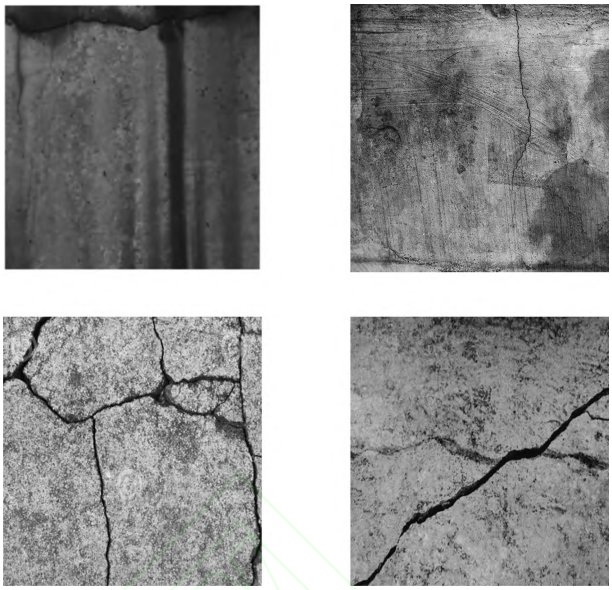
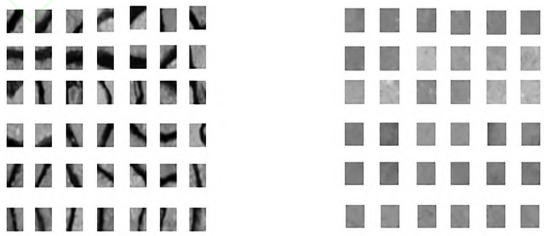


图 3　桥梁裂缝图像特点示意图

Fig. 3　Image characteristics of bridge cracks

数据集扩充之后, 数据集中的桥梁裂缝面元和桥梁背景面元如图 4 所示; 其中桥梁裂缝面元有效的表达了桥梁裂缝的局部结构信息, 而桥梁背景面元则表示除桥梁裂缝结构信息以外的其他任何信息和桥梁图像中有可能出现的噪声信息; 这些面元的分辨率均为 16×16 像素的小图像.



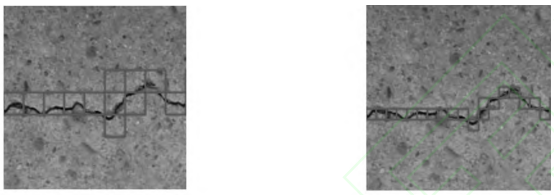
(a) 桥梁裂缝面元示意图 (a) Schematic diagram of bridge crack surface　(b) 桥梁背景面元示意图 (b) Schematic diagram of background surface

图 4　桥梁裂缝面元和背景面元示意图

Fig. 4　Schematic diagram of bridge crack surface and background surface

在深度学习中, 对于小图像的识别通常使用的是 CIFAR10[27] 模型. 但是, 此块 CIFAR10 模型并不适用于桥梁裂缝的检测, 其原因如下所示: 1) CIFAR10 模型是一种专门针对 32×32 分辨率图像的识别模型, 并不能识别 16×16 像素分辨率的小图像. 如果通过像拉普拉斯金字塔这样的图像处理技术强制将 16×16 像素分辨率的小图像拉伸为 32×32 像素分辨率的图像, 然后, 再用 CIFAR10 模型进行识别, 这样将会使得 CIFAR10 模型的识别

率较低, 达不到桥梁裂缝检测的要求. 原因在于, 把 16×16 像素分辨率的图像通过拉普拉斯金字塔向上重建为 32×32 像素分辨率图像的过程中, 该图像会出现一定程度的失真. 2) 如果直接将 CIFAR10 模型用于桥梁裂缝的检测, 即首先使用 32×32 像素大小的滑动窗口将桥梁裂缝图像切分为 32×32 像素大小的桥梁裂缝面元和桥梁背景面元; 然后使用这些图像训练一个基于 CIFAR10 的分类识别模型; 最后基于这个模型去检测桥梁裂缝. 那么, 这将会导致最终在整幅桥梁图像上检测出来的桥梁裂缝十分的不准确, 具体检测结果如图 5 中的 (a) 图所示, (b) 图为基于 DBCC 模型检测的结果. 3) 本文之所以提出 DBCC 模型而不使用 CIFAR10 模型, 不仅因为基于 DBCC 模型所检测出来的桥梁裂缝更加逼近真实的桥梁裂缝区域; 而且也因为桥梁裂缝面元的面积越小, 桥梁裂缝面元中所含有的噪声信息也越少. 但是, 桥梁裂缝面元的选取也不能无限制的小. 桥梁裂缝面元太小, 不仅不利于桥梁裂缝面元对于裂缝结构信息的表达, 也会导致识别网络模型难以构建、识别网络模型的网络深度过低, 这些最终都会严重降低模型的识别准确率. 这也是本文将桥梁裂缝面元选定为 16×16 像素分辨率的依据.



(a) CIFAR10 模型检测的结果示意图
(a) Detection results of CIFAR10 model

(b) DBCC 模型检测的结果示意图
(b) Detection results of DBCC model

图 5 CIFAR10 模型和 DBCC 模型检测结果的示意图

Fig. 5 Detection results of CIFAR10 and DBCC model

基于以上原因, 提出一种专门用于桥梁裂缝这种具有拓扑结构的、线性目标的检测识别模型十分必要. 本文在对 CIFAR10、AlexNet、GoogLeNet 等模型研究的基础之上, 提出了一种针对 16×16 像素分辨率的桥梁裂缝面元和桥梁背景面元的分类识别模型 DBCC (Deep-bridge-crack-classify) 模型.

2.2 DBCC 模型的构建

对于 DBCC 模型的具体构建, 本文将主要从以下几个方面展开: DBCC 模型的全模型网络结构、使用更小的输入图像、使用更大的卷积核、加深网络深度、每层使用更多的卷积核、添加 LRN 层、使用 Dropout 层. 这个顺序同时也是本文通过实验逐步构建 DBCC 模型的顺序.

2.2.1 DBCC 全模型

DBCC 模型使用了 4 层卷积层 (C1～C4), 3 层池化层 (P1～P3), 2 层全连接层 (FC1～FC2), 最后采用 Softmax 函数 (S) 作为损失函数. 在 C1、C4、P2、P3、FC1 后面各加一个激活函数 (RELU); 同时, 在第一卷积层后面添加 LRN 层, FC1 后面添加 Dropout 层. 最后一层输出桥梁裂缝面元和桥梁背景面元这两类面元相应的识别概率值. 卷积核数目从 32 开始, 每经过一次卷积层, 卷积核的数目翻倍, 直到 256 为止. 偏置项值初始化为 0.1. DBCC 模型的整个网络结构示意图如图 6 所示. 其中 In 表示输入的图像数据, C 表示卷积层, P 表示池化层, FC 表示全连接层, S 表示 Softmax 函数, Out 表示输出, Relu 表示激活函数, LRN 表示局部响应值归一化层, D 表示 Dropout 层.

2.2.2 使用更小的输入图像

由于桥梁裂缝是一种典型的线性目标, 并且具有一定的拓扑结构. 如果识别模型的输入图像选用 32×32 像素的分辨率, 即桥梁裂缝面元的分辨率选用 32×32 像素的分辨率, 那么最终由桥梁裂缝面元组成的桥梁裂缝将会十分的不准确, 如图 5 所示. 因

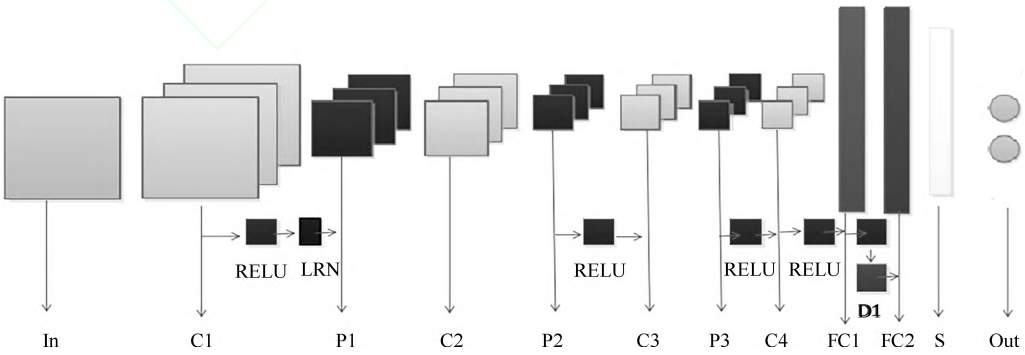


图 6 DBCC 模型的网络结构示意图

Fig. 6 Network structure of DBCC model

此, 本文选用了更小分辨率的输入图像作为桥梁裂缝面元和桥梁背景面元分类识别模型的输入图像, 并且在研究 CIFAR10 模型的基础之上, 提出了 DBCC-A 模型, 具体的模型构建参数如表 1 和表 2 中的第二行所示.

2.2.3　使用更大的卷积核

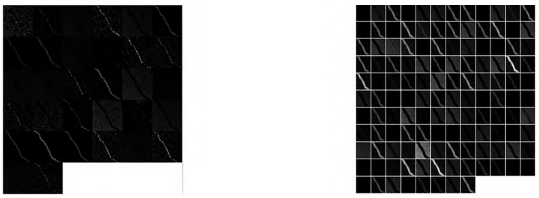
DBCC-A 模型输入图像的分辨率和 CIFAR10 模型输入图像的分辨率相比, 减小了一半. 因此, DBCC-A 模型各层的卷积核尺寸也按一定的比例进行了减小. 但是, 提出 DBCC-A 模型的目的主要是为了识别桥梁裂缝面元, 而使用更大的卷积核不仅有利于提取桥梁裂缝面元中裂缝的结构信息, 而且还有利于忽略掉和桥梁裂缝结构无关的其他噪声信息. 因此, 本文对 DBCC-A 模型做了进一步的改进, 并且提出了 DBCC-B 模型. 在 DBCC-B 模型中, 各卷基层使用了相对较大的、5×5 像素分辨率的卷积核, 并且为了保证识别模型的卷积层数不会随着卷积核分辨率的增大而减少, DBCC-B 模型为各卷基层的特征映射图添加了大小为了 2 的扩展边距. 其中 DBCC-B 模型具体的构建参数如表 1 和表 2 中的第三行参数所示.

2.2.4　加深网络深度

实验结果表明, 网络的深度在一定条件下, 越深结果越好[23, 29]. 因此, 针对本文中 16×16 像素分辨率的桥梁裂缝面元和桥梁背景面元, 为了尽可能的加深网络结构的深度, DBCC-B 模型为各卷基层的特征映射图添加了大小为 2 的扩展边距, 而且本文为了进一步加深网络结构的深度, 对 DBCC-B 模型中池化层的池化窗口做了进一步的缩小. 在此改进的基础之上, 提出了 DBCC-C 模型. 其具体构建的参数如表 1 和表 2 中的第四行参数所示.

2.2.5　每层使用更多的卷积核

在卷积层中, 每一个卷积核都可以被看成一个特征提取器, 卷积层中每一幅输出的特征映射图 (Feature map) 都可以被看成输入图像经过一个卷积核进行特征提取之后的结果, 但是通过对各卷积层的输出结果进行可视化对比可知, 并不是每一个卷积核都可以成功提取输入图像的特征, 从而得到有效的特征表达 (特征映射图). 如图 7 中的图 (a) 为 DBCC 模型第一卷积层的可视化结果, 图 (b) 为第四卷积层前 128 个特征映射图的可视化结果. 因此, 为了增强卷积层的特征表达能力, 对输入图像的特征进行充分的提取, 本文在 DBCC-C 模型的各卷积层中使用了更多的卷积核, 并且基于这一改进, 提出了 DBCC-D 模型. 其具体构建的参数如表 1 和表 2 中的第五行参数所示.



(a) 第一卷积层的可视化结果
(a) Visual results of the first convolutional layer

(b) 第四卷积层的可视化结果
(b)Visual results of the fourth convolutional layer

图 7　卷积层可视化结果示意图

Fig. 7　Visualization results of convolution layer

2.2.6　添加局部响应值归一化层 (LRN)

LRN 完成了一种“邻近抑制”操作, 对局部输入区域进行了归一化. 可以用于图像的明亮度矫正, 而桥梁裂缝图像由于光照、阴影等因素, 会出现图像亮度不均的问题, 因此, 本文在 DBCC−D 模型的第一卷积层之后添加了 LRN 层, 并且基于这一改进提出了 DBCC-E 模型. 其具体构建的参数如表 1 和表 2 中的第六行参数所示.

2.2.7　使用 Dropout 层

Dropout[28] 是指在训练模型时, 随机的让网络中某些隐含层的节点暂时不工作, 不工作的那些节点可以暂时的认为不是网络结构的一部分, 但是它们的权值保留了下来 (暂时不更新), 而在下次样本输入的时候, 随机的进行选择, 它们可能又可以工作. 所以, 每一次的样本输入, 都相当于随机选取了一个不同的网络结构进行训练, 但是这些不同的网络却共同训练出了共享的权值. 因此, Dropout 可以看作是不同学习模型之间组合的一种替代方法, 而使用不同的模型训练同一样本又是防止过拟合的一种方法, 因此, 对 DBCC-E 模型添加 Dropout 层, 可以有效的防止过拟合. 由于训练模型的数据集较小, 因此, 采用更大概率的 Dropout 进行补偿, Dropout 取值 0.55, 本文基于对 DBCC-E 模型的这一改进, 最终提出了 DBCC 模型. 其具体构建的参数如表 1 和表 2 中的第七行参数所示.

2.2.8　DBCC 模型的网络深度和卷积层数确定的依据

本文在构建 DBCC 模型的过程中, 除了依据以上的几点逐步构建出 DBCC 模型之外, DBCC 模型中各卷基层和各池化层的输入特征映射图和输出特征映射图的分辨率还必须满足式 (2) 所表述的计算关系, 这也是最终确定 DBCC 模型网络深度和卷积层个数最主要的依据之一.

$$\begin{cases} F_x^l = \dfrac{(F_x^{l-1}+2\times pad-K_x)}{S_x}+1 \\ F_y^l = \dfrac{(F_y^{l-1}+2\times pad-K_y)}{S_y}+1 \end{cases} \tag{2}$$

其中, $({F_x}^{l-1}, {F_y}^{l-1})$ 为前一卷积层或者池化层输入到本层的特征映射图; (K_x, K_y) 为本层卷积层卷积核的大小或者池窗口的大小; S_x 和 S_y 为卷积核或者池化窗口在特征映射图上滑动的步长; pad 为给当前的特征映射图添加的边界宽度; (F_x^l, F_y^l) 则为当前层的特征映射图经过卷积或者池化之后输出的特征映射图.

3 改进的窗口滑动算法

DBCC 深度学习模型实现了针对 16×16 像素分辨率这样小尺寸的桥梁裂缝面元图像和桥梁背景面元图像的分类识别, 但是如果想要最终检测出整幅桥梁裂缝图像中的桥梁裂缝, 还需要结合本文改进的窗口滑动算法. 本文改进的窗口滑动算法如图 8 所示.

如果直接使用传统的窗口滑动算法和 DBCC 模型结合检测桥梁裂缝, 则有可能检测出过多的桥梁噪声面元. 其原因在于, DBCC 模型最后使用了一个 Softmax 函数, 分别输出在本次识别过程中桥梁背景面元和桥梁裂缝面元的概率 $P_b(x)$ 和 $P_c(x)$, 然后, 对两者的概率进行比较, 具体的计算过程如式 (3) 所示. 其中 $f(x)$ 为 0, 表示当前的图像为桥梁背景面元图像, $f(x)$ 为 1, 表示当前的图像为桥梁裂缝面元图像.

$$f(x) = \begin{cases} 0, & p_b(x) > p_c(x) \\ 1, & p_b(x) \le p_c(x) \end{cases} \tag{3}$$

如果只根据 $P_b(x)) \le P_c(x)$, 就认为当前的桥梁面元图像为桥梁裂缝面元图像, 那么在整幅桥梁裂缝图片识别的时候, 有可能将某些桥梁噪音面元误识别为桥梁裂缝面元. 因为很多桥梁噪音面元最终识别出来的概率也有可能大于桥梁背景面元的概率. 针对这一问题, 对窗口滑动算法进行了改进. 具体做法是对式 (3) 进行了改进, 改进后的公式如式 (4) 所示:

表 1 DBCC 模型的输入层至第二池化层各层的具体模型构建参数

Table 1 Modeling parameters from the input layer to the second pool layer of the DBCC model

模型	输入层	卷积层 1	Max-Pooling1	卷积层 2	Ave-Pooling2
CIFAR10	$32 \times 32 \times 3$	Conv5-1-2-32	MP3-2-0-32	Conv5-1-2-32	AVE3-2-0-32
DBCC-A	$16 \times 16 \times 3$	Conv3-1-1-32	MP3-2-0-32	Conv3-1-1-32	AVE3-2-0-32
DBCC-B	$16 \times 16 \times 3$	Conv5-1-2-32	MP3-2-0-32	Conv5-1-2-32	AVE3-2-0-32
DBCC-C	$16 \times 16 \times 3$	Conv5-1-2-32	MP2-2-0-32	Conv5-1-2-32	AVE2-2-0-32
DBCC-D	$16 \times 16 \times 3$	Conv5-1-2-32	MP2-2-0-32	Conv5-1-2-64	AVE2-2-0-64
DBCC-E	$16 \times 16 \times 3$	Conv5-1-2-32-LRN	MP2-2-0-32	Conv5-1-2-64	AVE2-2-0-64
DBCC	$16 \times 16 \times 3$	Conv5-1-2-32-LRN	MP2-2-0-32	Conv5-1-2-64	AVE2-2-0-64

表 2 DBCC 模型的第三卷积层至输出层各层的具体模型构建参数

Table 2 Modeling parameters from the third volume accumulated layer to the output layer of the DBCC model

模型	卷积层 3	Ave-Pooling3	卷积层 4	FC1	Dropout 层	FC2	输出层
CIFAR10	Conv5-1-2-64	AVE3-2-0-64	—	2	—	—	Softmax
DBCC-A	Conv3-1-1-64	AVE3-2-0-64	—	2	—	—	Softmax
DBCC-B	Conv5-1-2-64	AVE3-2-0-64	—	2	—	—	Softmax
DBCC-C	Conv5-1-2-64	AVE2-2-0-64	Conv2-1-0-64	2	—	—	Softmax
DBCC-D	Conv5-1-2-128	AVE2-2-0-128	Conv2-1-0-256	128	—	2	Softmax
DBCC-E	Conv5-1-2-128	AVE2-2-0-128	Conv2-1-0-256	128	—	2	Softmax
DBCC	Conv5-1-2-128	AVE2-2-0-128	Conv2-1-0-256	128	Dropout	2	Softmax

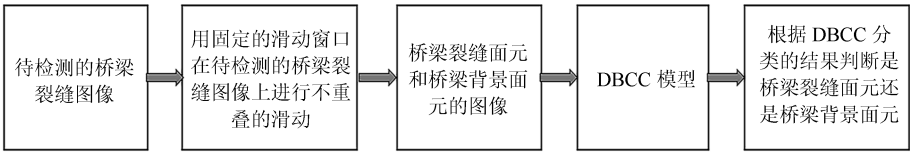


图 8 改进的窗口滑动算法示意图

Fig. 8 Improved window sliding algorithm

$$f(x)=\begin{cases}0, & p_b(x)>p_c(x)\\ 1, & p_b(x)\le p_c(x), p_c(x)>T_P\end{cases} \tag{4}$$

其中 Tp 为一个概率区分阈值. 根据实验, Tp 一般可以取 (0.90 0.99) 之间的数.

4　算法的加速策略

为了降低算法的时间复杂度, 加速算法的处理速度, 满足桥梁裂缝检测实时处理的要求, 必须采用多种措施来提高算法的执行效率. 本文主要采用图像金字塔和 ROI 区域相结合的策略对本文的桥梁裂缝检测的算法进行加速. 由 DBCC 模型结合改进的窗口滑动算法对桥梁裂缝的检测过程可知, 算法的运行时间主要取决于检测目标图像的大小. 针对这一问题, 提出一种基于图像金字塔和 ROI 区域相结合的加速策略. 即首先针对要识别的桥梁裂缝图像构建图像金字塔, 如图 9 所示. 然后, 在低分辨率的图像上使用检测算法对桥梁裂缝进行检测, 同时对识别出来的桥梁裂缝面元的横纵坐标进行排序, 求出包含裂缝的矩形区域的左上角坐标和右下角坐标, 然后将求出来的坐标代入式 (5) 确定在高分辨率图像上包含裂缝的矩形区域, 并且将这一矩形区域设为 ROI 区域. 最后, 使用检测算法对 ROI 区域的桥梁裂缝图像进行检测. 具体的算法加速策略可以参考图 9.

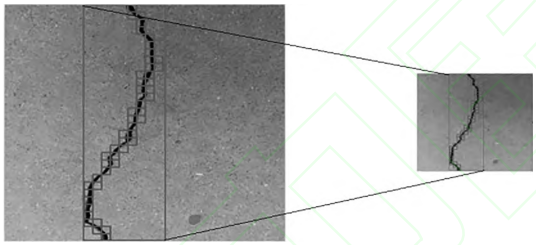


图 9　图像金字塔和 ROI 加速策略示意图

Fig. 9　Schematic diagram of image pyramid and ROI acceleration strategy

其中, 式 (5) 的具体公式如下所示, 其中坐标 (x_l, y_l) 为低分辨率图片上确定的位置坐标, (x_h, y_h) 为高分辨率图片确定的位置坐标. n 代表桥梁裂缝图片通过高斯图像金字塔向下采样的次数.

$$\begin{cases}x_h = x_l \times 2^n\\ y_h = y_l \times 2^n\end{cases} \tag{5}$$

5　基于桥梁裂缝面元的裂缝提取和定位算法

桥梁裂缝图像经过本文中上述算法的处理, 其结果已经十分逼近桥梁裂缝的真实区域和面积, 并且已经成功的将桥梁裂缝图像中的噪声隔离于桥梁裂缝之外, 如图 10 所示. 但是, 为了更加准确的提取桥梁裂缝和确定桥梁裂缝在桥梁图像中的位置, 本文提出了一种基于桥梁裂缝面元的桥梁裂缝提取和定位算法. 该算法可归纳为以下三个步骤:

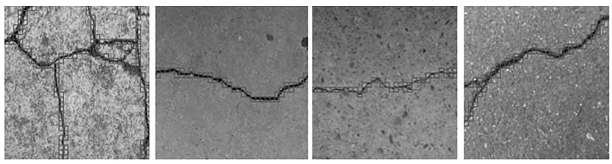


图 10　DBCC 模型桥梁裂缝检测结果的示意图

Fig. 10　Schematic diagram of DBCC model bridge crack detection results

1) 采用已经训练好的 DBCC 模型和改进的窗口滑动算法对桥梁裂缝图像进行检测, 识别出桥梁裂缝图像中所有的桥梁裂缝面元, 并且将识别出来的桥梁裂缝面元使用相同分辨率的标识框标记出来, 最终桥梁裂缝检测的结果如图 11 中的 (b) 图所示;

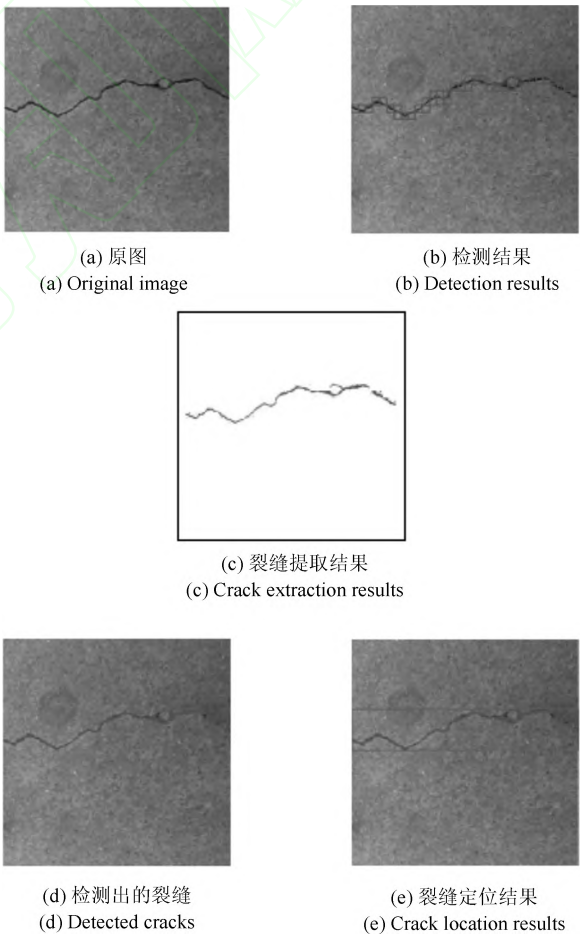


图 11　基于桥梁裂缝面元的裂缝提取和定位算法流程示意图

Fig. 11　Flow chart of crack extraction and location algorithm based on bridge crack surface element

2) 使用 Otsu 阈值化算法求解出桥梁裂缝图像全局的分割阈值 T, 并且根据这个求解出来的阈值 T 对每一个桥梁裂缝面元进行固定化阈值分割; 由于现在的桥梁裂缝面元已经十分逼近桥梁裂缝, 并且已经将干扰噪声排除在外, 因此, 基于桥梁裂缝面元和求解出来的阈值 T 可以十分准确的提取出桥梁裂缝面元中的桥梁裂缝, 所有基于桥梁裂缝面元提取出的桥梁裂缝最终组成整幅图像的裂缝提取结果, 如图 11 中的 (c) 图所示;

3) 本步对上一步骤中提取的桥梁裂缝中所有的像素点的 X 坐标和 Y 坐标分别进行排序, 将最小的 X 坐标和 Y 坐标组成的坐标点作为裂缝区域的左上角点坐标, 将最大的 X 坐标和 Y 坐标所组成的坐标点作为裂缝区域的右下角点坐标; 最后, 标记出由这两个坐标点所确定的矩形区域, 这个区域即为桥梁裂缝的位置.

6 实验结果与分析

本文采用真实的桥梁裂缝图像数据进行试验, 桥梁裂缝图像是由大疆无人机 Phantom 4 pro 自带的 CMOS 面阵相机采集的, 具体的采集方法是让无人机在桥梁裂缝的附近进行悬停, 然后通过无人机上的云台调整面阵相机的姿态, 使得相机的镜头平行于桥梁裂缝的表面, 并且要求相机的镜头距离桥梁裂缝的表面 30 cm, 调整好相机的姿态和距离后, 让无人机从悬停状态转换为沿着裂缝方向平稳飞行, 连续拍照. 本文一共采集了 2 000 张桥梁裂缝图像, 并且将这 2 000 张桥梁裂缝图像分为两个集合, 将这两个集合分别称为 A 集合和 B 集合. 其中,A 集合中的 1 000 张图像结合数据集人工扩增方法用于构建训练 DBCC 模型的训练集, 验证集; B 集合中的 1 000 图像用以构建本文算法的测试集, 用于测试本文算法中的各项参数以及算法的性能, 并且 B 集合中的 1 000 张图像在实验结果和分析部分统一从原来 1 024×1 024 分辨率的图像下采样为 512×512 分辨率的图像. 本文算法的程序基于主流的深度学习开源框架 Caffe 和计算机视觉开源库 OpenCV, 使用 C/C++、python 语言开发; 程序的运行环境为 Ubuntu14.04, CPU3.3 GHz, RAM 8 G.

为了验证本文提出算法的有效性和准确性, 本文分别设计了八组对比实验, 用以验证本文算法的各个方面. 其中, 第一组实验用于测试数据集人工扩增方法对于 DBCC 模型识别准确率的影响. 第一组实验的设计步骤分为三步. 首先, 直接使用 A 集合中 1 000 张桥梁裂缝图像、不经过数据集人工扩增方法构建 DBCC 模型的训练集、验证集, 用以训练 DBCC 模型; 然后, 使用 A 集合中 1 000 张图像, 并使用数据集人工扩增方法对这 1 000 图像进行扩增, 使用扩增之后的数据集构建训练集、验证集, 用以训练 DBCC 模型. 最后, 再从 B 集合中随机的选取数幅桥梁裂缝图像并将其切分为 16×16 像素大小的小图像, 随机的选取 500 张桥梁裂缝面元图像用以测试训练好的 DBCC 模型. 测试的具体结果如表 3 所示.

由第一组实验结果可知, 利用数据集人工扩增方法扩展之后的数据训练的 DBCC 模型对于桥梁裂缝面元识别的准确率得到了极大的提高; 没有经过数据集扩增的数据训练的 DBCC 模型识别准确率之所以低, 其主要原因是因为 DBCC 模型没有得到充分的训练, 出现了欠拟合现象.

第二组实验用于对比 CIFAR10 模型和 DBCC 模型对于桥梁裂缝最终检测结果的影响. 第二组实验包含三个小实验. 实验 1, 先使用 32×32 像素分辨率的滑动窗口扩增的数据集训练 CIFAR10 模型, 再使用训练好的 CIFAR10 模型对于 B 集合中的桥梁裂缝图像进行检测. 实验 2, 使用 DBCC 模型对于 B 集合中的桥梁裂缝图像进行检测; 实验 3, 先使用 16×16 像素分辨率的滑动窗口对桥梁裂缝图像进行滑动, 然后使用拉普拉斯图像金字塔, 将每一个 16×16 像素的小图像面元向上重建为 32×32 像素的小图像面元, 最后将重建的图像面元送入 CIFAR10 模型进行识别. 三种方式的检测结果如图 12 所示.

其中图 12 的第一行为实验 1 的检测结果; 第二行为实验 2 的检测结果; 第三行为实验 3 的检测结果. 由实验结果可知, 基于 DBCC 模型的桥梁裂缝检测结果十分逼近桥梁裂缝的真实区域, 也能够较好的表达桥梁裂缝的拓扑结构; 而直接使用 CIFAR10 模型的检测结果, 相对于 DBCC 的检测结果来说, 十分的不准确. 实验 3 的结果说明, 如果先

表 3 人工数据集扩增方法对于 DBCC 模型识别准确率的影响

Table 3 Effect of artificial data set amplification on recognition accuracy of DBCC model several methods for uneven illumination document images

桥梁裂缝面元的总数	有无数据集的扩增	DBCC 模型正确识别数	准确率 %
500	无	353	70.6
500	有	489	97.8

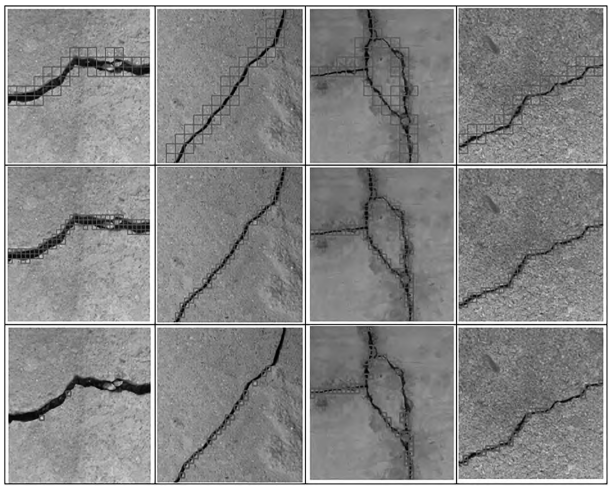


图 12　CIFAR10 模型和 DBCC 模型对于桥梁裂缝检测结果的对比

Fig. 12　Comparison between CIFAR10 model and DBCC model for bridge crack detection

用 16×16 的滑动窗口滑动桥梁裂缝, 再将窗口覆盖下的小图像面元通过拉普拉斯金字塔这样的图像处理方法强制拉伸为适合 CIFAR10 模型识别的面元, 这将会导致 CIFAR10 模型的识别率较低. 因此, 提出一个专门用于桥梁裂缝检测的深度学习模型是极其必要的.

第三组实验用于测试 CIFAR10、DBCC-A、DBCC-B、DBCC-C、DBCC-D、DBCC-E、DBCC 模型对于桥梁裂缝面元识别的准确率以及 CIFAR10、DBCC 模型在不同尺寸滑动窗口下对于桥梁裂缝面元的识别率. 其中各种模型的训练集使用的是 A 集合经过人工数据集扩增之后的训练集. 测试集合是随机从 B 集合中选取数幅图像, 使用表 4 中每行模型对应的相应尺寸的滑动窗口切分, 然后从切分的集合中随机的选取 1 000 张桥梁裂缝面元用于测试各种模型. 最终具体的实验和实验结果如表 4 所示.

通过 DBCC-A 模型和 DBCC-B 模型的结果对比可知, 使用更大的卷积核有利于识别模型准确率的提升; 由 DBCC-B 和 DBCC-C 的结果对比可知, 虽然识别模型的准确率确实有提升, 但是提升较小, 这主要是因为 DBCC-C 相对于 DBCC-B 模型的网络深度加深较小所致; 由 DBCC-C 和 DBCC-D 的结果对比可知, 识别模型的准确率提升较大, 这说明在一定条件下, 增加每个卷积层的卷积核数量, 有利于识别模型提取到更多有关桥梁裂缝的特征信息, 这将会增加识别模型对于桥梁裂缝面元的识别能力; 由 DBCC-D 和 DBCC-E 的结果对比可知, 为第一卷积层增加 LRN 确实有利于准确率的提升; 由 DBCC-E 和 DBCC 在同样尺寸滑动窗口下的识别率对比可知, 为模型添加 Dropout 层, 同样也可以提升模型的识别准确率. 由第 0 行和第 8 行的结果可知, 当滑动窗口尺寸取值过小时, 无论是 CIFAR10 模型还是 DBCC 模型均不能对滑动窗口覆盖下的桥梁裂缝面元图像进行很好的识别, 这主要是因为较小的滑动窗口覆盖下的桥梁裂缝面元图像包含的像素信息太少, 已经不足以表达裂缝的结构信息, 从而导致此情况下的识别率极其低下; 由第 1 行、第 2 行、第 9 行的结果对比可知, 如果不构建新的模型, 而是使用 CIFAR10 模型直接对 16×16 像素大小滑动窗口覆盖下的桥梁裂缝面元图像进行识别, 其识别率将会十分的不理想, 这主要是因为在将 16×16 像素大小的桥梁裂缝面元图像通过拉普拉斯图像金字塔向上重建为 32×32 像素大小、适合 CIFAR10 模型的识别图像时, 会丢失桥梁裂缝的一部分结构信息所导致的; 由第 2 行、第 9 行的

表 4　各模型对于桥梁裂缝面元识别的准确率

Table 4　Accuracy of each model for bridge crack surface element identification

行号	模型	裂缝面元的总数	裂缝面元规格/像素	正确识别的裂缝面元数/幅	识别的准确率/%
0	CIFAR10	1 000	8×8	34	3.4
1	CIFAR10	1 000	16×16	347	34.7
2	CIFAR10	1 000	32×32	976	97.6
3	DBCC-A	1 000	16×16	954	95.4
4	DBCC-B	1 000	16×16	958	95.8
5	DBCC-C	1 000	16×16	960	96.0
6	DBCC-D	1 000	16×16	970	97.0
7	DBCC-E	1 000	16×16	973	97.3
8	DBCC	1 000	8×8	48	4.8
9	DBCC	1 000	16×16	979	97.9

对比结果可知, 虽然两者识别准确率几乎持平, 但是由图 5 和图 12 的对比可知, 第 2 行情况下, 基于 CIFAR10 模型的桥梁裂缝检测结果远不如第 9 行情况下 DBCC 模型的桥梁裂缝检测结果, 而且由于第 2 行对应的滑动窗口为 32×32 像素, 这也就意味着最终的检测结果中将会包含更多的噪声, 而基于第 9 行 DBCC 模型的检测结果, 由于是 16×16 像素大小的滑动窗口, 这种情况下的检测结果不仅更加逼近真实的桥梁裂缝, 而且对于噪声的排除和抑制能力更强, 这说明本文提出的 DBCC 模型在针对桥梁裂缝检测这一特定问题上的性能完全优于 CIFAR10 模型, 也进一步说明, 针对桥梁裂缝检测, 提出一种新的识别模型是十分重要的.

第四组实验用于测试改进的窗口滑动算法在对桥梁裂缝图像进行桥梁裂缝检测时, 概率区分阈值 T_p 对于桥梁裂缝检测效果的影响. 具体的检测结果如图 13 所示.

由图 13 可知, 当概率区分阈值 T_p 的取值较小时, 会有大量的桥梁面元噪声被检测出来. 根据实验可知: 当 T_p 的取值在 0.9 到 0.99 之间时, 桥梁裂缝检测的结果可以满足检测的要求. 因此, 在本文的所有实验中, 令 T_p 的取值为 0.96.

第五组实验用于测试本文提出的算法加速策略对于桥梁裂缝检测算法处理速度的影响. 具体的实验分为两个独立的实验. 实验 1, 不使用算法的加速策略, 直接使用本文提出的桥梁裂缝检测算法对桥梁图像进行裂缝检测; 实验 2, 结合算法的加速策略, 对桥梁图像进行裂缝检测. 这两个实验所使用图像的分辨率均为 1 024×1 024 像素. 两种方法的耗时如表 5 所示.

通过对第五组实验结果的对比可知, 在本文中提出的算法加速策略可以极大的提高桥梁裂缝检测算法的检测速度.

表 5 算法加速策略对于本文识别算法的影响

Table 5 Effect of algorithm acceleration strategy on the recognition algorithm in this paper

桥梁裂缝图像的编号	未采用加速策略耗时/s	采用加速策略耗时/s
1	16.976065	3.345511
2	17.332235	3.324066
3	16.834522	3.295553
4	17.793048	3.500678
5	17.123332	3.349923

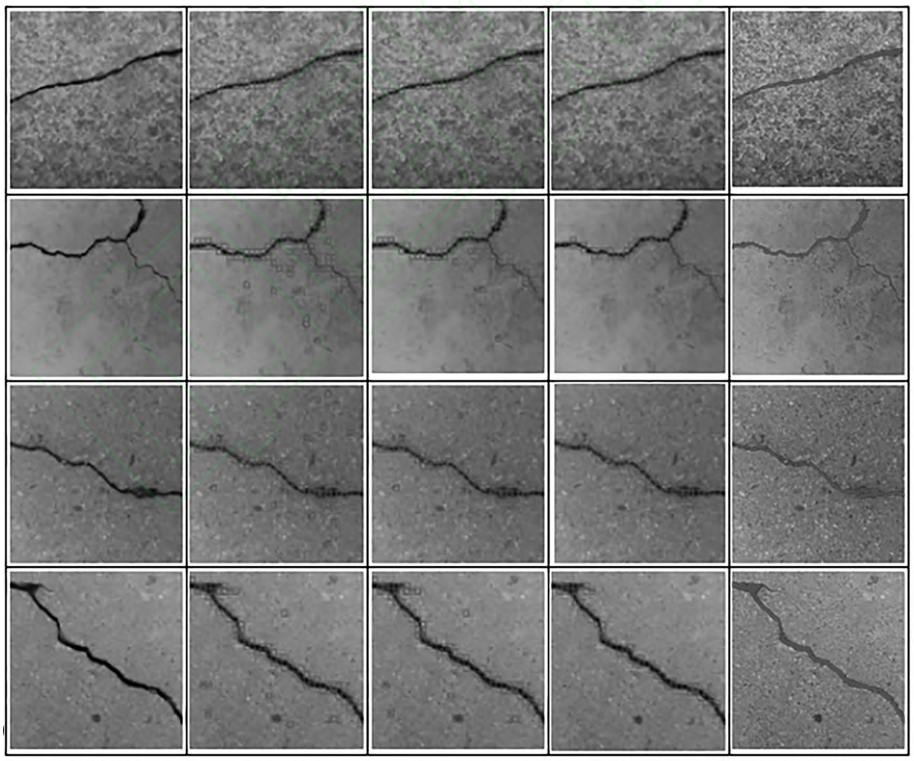


图 13 概率区分阈值 T_p 对于桥梁裂缝识别效果的影响

Fig. 13 Effect of probability discrimination threshold T_p on bridge crack identification

第六组实验用于测试基于桥梁裂缝面元的桥梁裂缝提取算法对于桥梁裂缝检测最终结果的影响. 首先, 本文从 1 000 张用于桥梁裂缝检测的五类不同材质的图像集合中, 随机的选取 100 张不同背景纹理的桥梁裂缝图像. 然后, 使用这些桥梁裂缝图像对该算法进行测试, 测试的部分结果如图 14 所示.

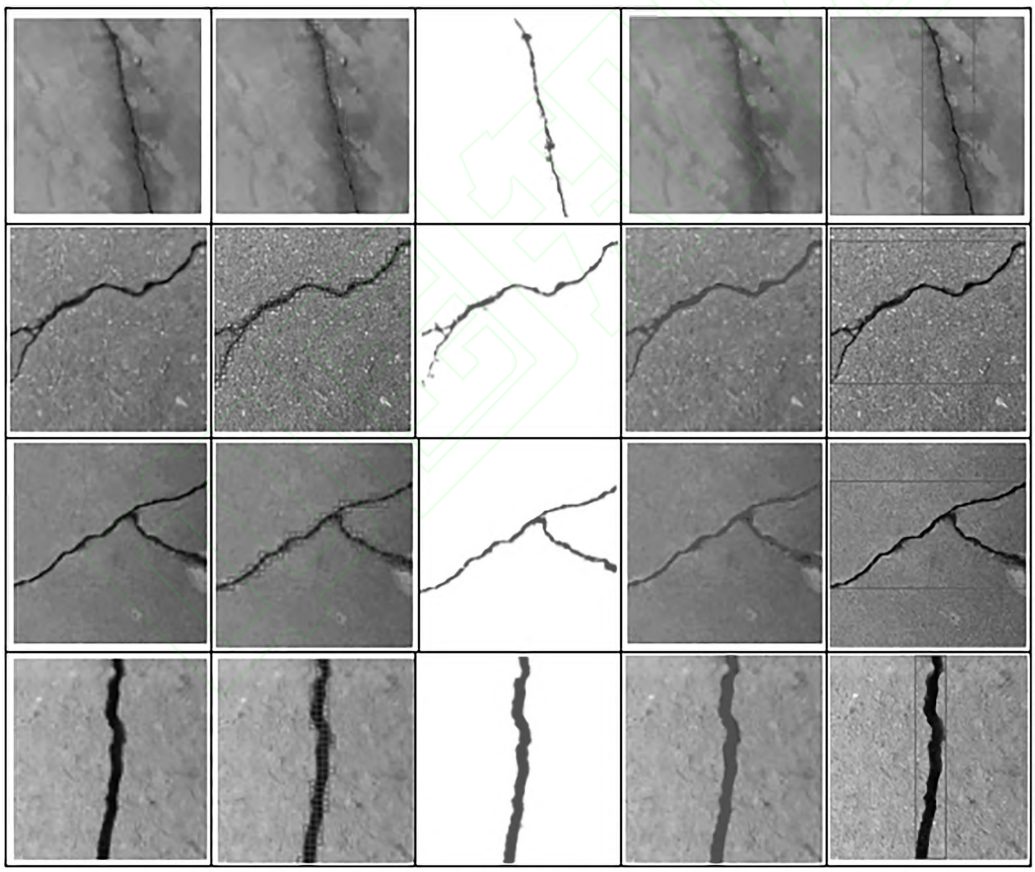
通过对图 14 的第二列图像和第四列图像的对比可知, 基于桥梁裂缝面元的裂缝提取算法将会使得最终的桥梁裂缝检测结果更加准确. 通过对第一列的图像和第三列的图像对比可知, 基于桥梁裂缝面元的裂缝提取算法结合本文中提出的其他算法, 最终可以比较准确的提取出桥梁裂缝图像中的裂缝. 图 14 中的第五列图像则为根据提出来的裂缝对图像中的裂缝定位的结果; 裂缝所在的位置, 即为矩形区域所标注的位置. 另外, 表 6 给出了桥梁裂缝所在矩形区域的坐标点, 本文就是利用这两个坐标点确定的桥梁裂缝位置. 对图 14 中的第一列图像从上到下进行编号, 编号为 1-4, 最终的坐标点位置如表 6 所示.

表 6　桥梁裂缝定位的位置坐标

Table 6　Position coordinates of bridge crack location

图像的编号	左上角点坐标/像素	右下角点坐标/像素
1	(239, 0)	(381, 512)
2	(0, 98)	(512, 370)
3	(0, 33)	(512, 385)
4	(218, 0)	(294, 512)

第七组实验用于测试各种裂缝检测算法对于最终裂缝定位准确度的影响. 具体的实验步骤如下所示: 首先, 本文从 1 000 张用于桥梁裂缝测试实验的图像集合中, 随机的选取 500 张桥梁裂缝图像, 并且在这些图像上, 手动的标记出裂缝的外接矩形并记录外接矩形的左上角点坐标和右下角点的坐标, 如



(a) 原图 (a) Original image　(b) DBCC检测结果 (b) Detection results of DBCC model　(c) 裂缝提取的结果 (c) Crack extraction results　(d) 提取的裂缝在原图上的标注 (d) Stamping of crack extraction results on original image　(e) 裂缝定位结果 (e) Crack location results

图 14　基于桥梁裂缝面元的桥梁裂缝提取算法对于检测结果的影响

Fig. 14　Influence of bridge crack extraction algorithm based on bridge crack surface element on detection

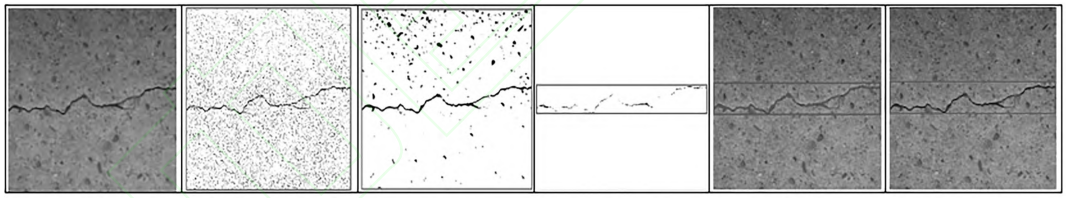
图 15 中的第六列图像所示; 然后, 使用不同的裂缝检测和提取算法提取出桥梁的裂缝; 最后, 使用本文的裂缝定位算法基于上一步的结果, 对图像中的裂缝进行定位; 图 15 给出了定位实验中具有代表性的一组定位结果.

在第六组和第七组实验的基础上, 为了对本文提出的桥梁裂缝定位算法进行更加深入的分析和评价, 本文引入了桥梁裂缝定位准确度指数 S, 用于对裂缝定位的效果进行量化和分析. 其中, S 表明了裂缝真实的外接矩形和使用裂缝定位算法定位出来的外接矩形的偏差度; 若两个外接矩形所定位的裂缝位置完全重合, 则 S 为 0; 否则 S 越大, 表明裂缝定位算法定位的裂缝位置越不准确; (x, y) 和 (x', y') 为用裂缝定位算法定位出来的裂缝外接矩形的左上角点坐标和右下角点坐标; (r, c) 和 (r', c') 为在选取出来的 500 张桥梁裂缝图像上手动标记出来的桥梁裂缝实际外接矩形的左上角点坐标和右下角点坐标. 桥梁裂缝定位准确度指数 S 的具体公式如式 (6) 所示; 各种裂缝检测算法在定位准确度 S 一定的条件下, 定位出来的裂缝位置在准确度 S 范围内的准确率如表 7 所示.

$$S=\frac{\sqrt{(x-r)^2+(y-c)^2}+\sqrt{(x'-r')^2+(y'-c')^2}}{2} \tag{6}$$

通过量化实验可知, 如果设定定位算法定位出来的裂缝外接矩形与真实裂缝外接矩形的偏差度保持在 10 个像素以内认为定位准确, 则基于各检测算法的裂缝定位合格率为表 7 中的第二列数据所示; 由表 7 可知, 当准确度 S 的容忍越大, 则基于各个检测算法的定位合格率递增; 但是, 观察整个量化实验的数据可知: 基于本文提出的桥梁裂缝检测算法最终裂缝定位的合格率最高, 并且算法运行时间也最短. 基于本文提出的桥梁裂缝检测算法之所以会取得比较好的裂缝定位的准确度, 其原因在于: 本文在桥梁裂缝检测的过程中, 在桥梁裂缝面元的识别阶段, 已经排除了大量的干扰噪声, 在桥梁裂缝的检测阶段已经将桥梁裂缝所在的区域锁定在一个比较靠近桥梁裂缝位置的区域, 而后续的桥梁裂缝提取算法和桥梁裂缝定位算法都是在这一很小的桥梁裂缝区域内进行的运算, 因此, 本文所提出的桥梁裂缝检测算法是一种性能可靠、稳定的桥梁裂缝检测算法, 这对于桥梁裂缝的检测、提取和定位来说都是及其重要的.

第八组实验用于测试本文算法和其他主流裂缝检测算法的对比. 为了说明本文算法相比于其他主流的裂缝检测算法, 具有更好的识别效果和更强的泛化能力. 该组实验从 B 集合的 1 000 张桥梁裂缝图像中选取了五类不同材质不同背景纹理的桥梁裂缝图像, 且特意选取为带有复杂背景和严重噪声的桥梁裂缝图像, 如图 16 中的第二行至第五行图像所示. 然后, 利用本文提出的算法与迭代阈值分割算法[3]、FFA 算法[11]、最小路径选择算法[8] 相对比, 进一步说明本文算法的性能.



(a) 原图 (a) Original image
(b) 迭代阈值分割算法 (b) Iterative threshold segmentation algorithm
(c) FFA 算法 (c) FFA algorithm
(d) 最小路径择算法 (d) Minimum path selection algorithm
(e) 本文算法 (e) Our algorithm
(f) 手动测量和标记的桥梁裂缝的外接矩形 (f) Manually measuring and marking the outer rectangle of a bridge crack

图 15 各种裂缝检测算法对于桥梁裂缝定位准确度的影响

Fig. 15 Influence of various crack detection algorithms on accuracy of bridge crack location

表 7 桥梁裂缝定位准确度的量化分析

Table 7 Quantitative analysis of location accuracy of bridge cracks

算法	S<10 像素的准确率	S<20 像素的准确率	S<30 像素的准确率	平均运行时间/s
迭代阈值分割算法[3]	10.4 %	20.0 %	27.4 %	0.998
FFA 算法[11]	13.4 %	27.1 %	34.8 %	3.976
最小路径选择算法[8]	17.6 %	37.6 %	61.0 %	11.493
本文算法	46.6 %	67.4 %	84.4 %	3.468

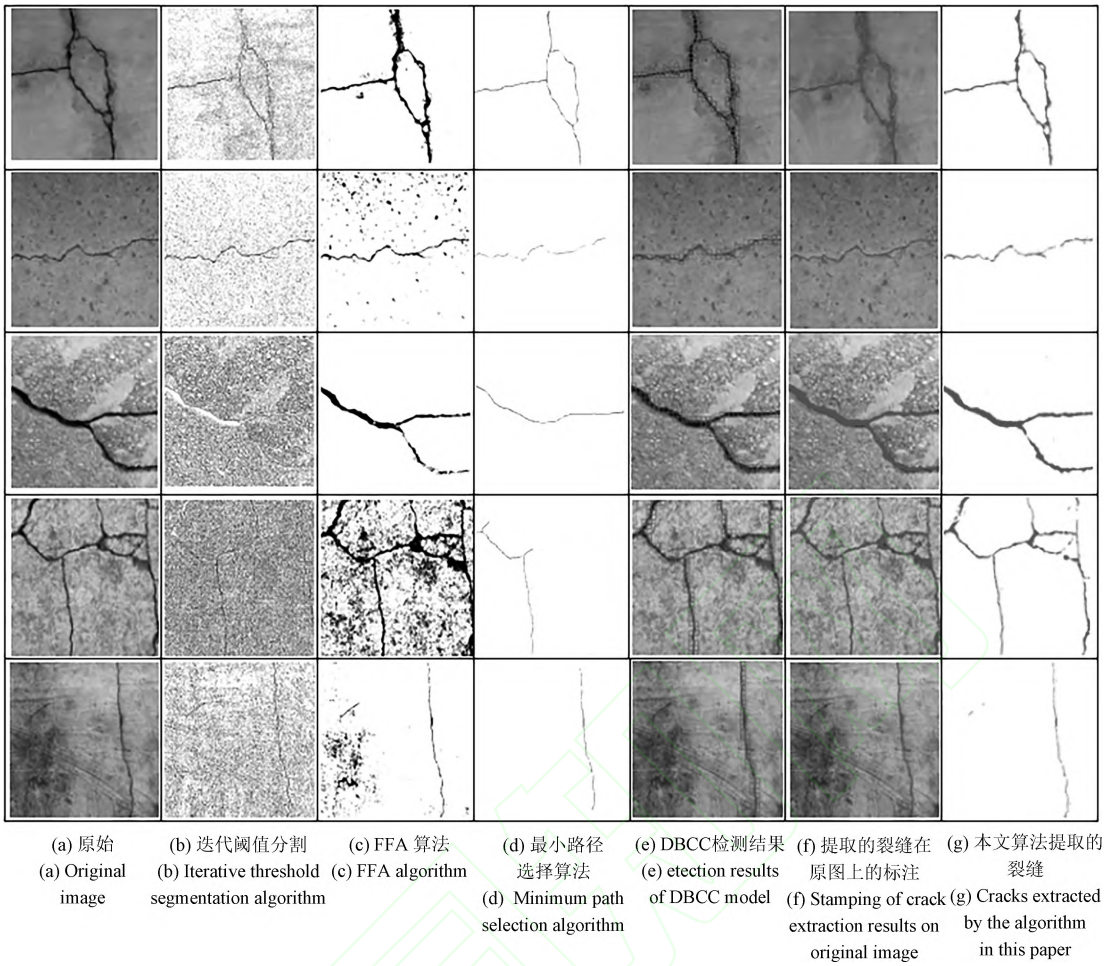


图 16　主流裂缝检测算法和本文算法对于桥梁裂缝检测的效果图

Fig. 16　Detection results of the main stream crack detection algorithm and our algorithm for bridge cracks image

由实验结果可知, 迭代阈值分割算法的检测结果会产生大量的噪声, 并且这些噪声几乎覆盖了桥梁裂缝. FFA 算法虽然对于某些桥梁裂缝图像的裂缝提取效果不错. 但是, 当图像背景变得复杂的时候, 在提取裂缝的同时也会产生大量的噪声, 如图 16 的第三列图像所示. 基于最小路径选择的裂缝提取算法, 在大多数情况下可以比较准确的提取出裂缝, 但是, 该算法最大的缺点就是有可能提取出来的桥梁裂缝不完整, 如图 16 中的第三行、第四行的第四列的图像所示; 而且, 该算法提取出来的裂缝也不能准确的表达裂缝的宽度. 但是, 通过图 16 中的第五列到第七列图像可知, 本文提出的桥梁裂缝检测算法可以很好的对不同类型的桥梁裂缝图像进行裂缝的检测和裂缝的提取, 提取出来的桥梁裂缝也能够比较准确的表达裂缝的宽度. 即使在面对背景复杂、噪声严重的桥梁裂缝图像时, 本文提出的算法也能够很好的完成桥梁裂缝的检测和提取, 如图 16 的第五行、第六列和第七列图像所示.

在上述实验结果比较的基础之上, 为了对本文提出的桥梁裂缝检测算法和桥梁裂缝提取算法进行更加深入的分析和评价, 本文引入了裂缝准确度指数 Pre 和裂缝召回率指数 Rec 用于对裂缝的检测和提取效果进行量化的分析和评价. 其中, 裂缝准确度指数 Pre 用于描述准确被检测提取出来的裂缝区域的像素数量 TP 占被检测提取出来的像素数量 $(TP + FP)$ 的比例; 裂缝召回率指数 Rec 用于描述被正确检测提取出来的裂缝区域的像素数量 TP 占应该被检测提取出来的裂缝区域像素数量的比例. 裂缝准确度指数 Pre 和裂缝召回率指数 Rec 具体的计算公式如下面的式 (7) 和式 (8) 所示.

$$Pre = \frac{TP}{TP + FP} \tag{7}$$

$$Rec = \frac{TP}{TP + FN} \tag{8}$$

式 (7) 和式 (8) 中的 TP 代表被正确检测提取出来的裂缝区域像素的数量, FP 代表被误判为裂缝区域像素的数量, FN 代表属于裂缝区域的像素但是

没有被检测出来的像素的数量. 图 16 中、第 1 行到第 5 行的 5 幅桥梁裂缝图像所对应的迭代阈值分割算法、FFA 算法、最小路径选择算法以及本文所提的桥梁裂缝检测提取算法的量化分析和运行时间如表 8 所示.

通过量化实验可知, 虽然迭代阈值分割算法在时间效率上十分具有优势, 但是 Pre 指数和 Rec 指数十分低下; FFA 算法和最小路径选择算法虽然在某些情况下, 效果较好, 比如第 1 幅图像, 但是在大多数情况下, 其 Pre 指数和 Rec 指数相对来说不是很理想; 而本文提出的桥梁裂缝检测和提取算法不仅在时间效率上表现出相对较好的结果, 尤其是在 Pre 指数和 Rec 指数上效果稳定且很理想.

通过以上的实验和量化分析结果对比, 说明本文提出的桥梁裂缝检测算法和其他主流的裂缝检测算法相比, 具有更好的识别效果和更强的泛化能力.

本文最终对 B 集合中的 1 000 张不同背景纹理、不同材质的桥梁裂缝图像进行了检测, 下面给出了基于本文算法进行桥梁裂缝检测的部分结果, 具体的桥梁裂缝检测结果如图 17 所示.

7 结论

本文提出了一种基于深度学习的桥梁裂缝检测算法. 讨论了桥梁裂缝数据集的人工扩增方法, 详细介绍了本文提出的 DBCC 模型和对窗口滑动算法的改进, 同时采用一定的加速策略对桥梁裂缝检测算法的执行时间进行了一定的优化. 实验结果表明, 和传统的裂缝检测算法相比, 本文提出的算法具有更好的识别效果和更强的泛化能力.

表 8 桥梁裂缝检测提取算法的量化分析对比

Table 8 Quantitative analysis and comparison of bridge crack detection and extraction algorithm

算法	迭代阈值分割算法	FFA 算法	最小路径选择算法	本文算法
评价指标	Pre/Rec	Pre/Rec	Pre/Rec	Pre/Rec
第 1 幅图像	15.7 %/16.3 %	89.3 %/91.5 %	90.3 %/91.5 %	97.5 %/98.7 %
第 2 幅图像	12.5 %/9.5 %	67.5 %/83.5 %	73.5 %/80.4 %	92.39 %/94.6 %
第 3 幅图像	5.6 %/5.57 %	90.5 %/90.3 %	63.3 %/64.39 %	94.5 %/96.3 %
第 4 幅图像	7.8 %/7.5 %	18.3 %/21.7 %	35.7 %/20.3 %	91.31 %/92.5 %
第 5 幅图像	10.1 %/13.4 %	73.2 %/63.7 %	91.2 %/93.5 %	91.6 %/90.4 %
平均运行时间/s	0.936	3.896	11.314	3.357

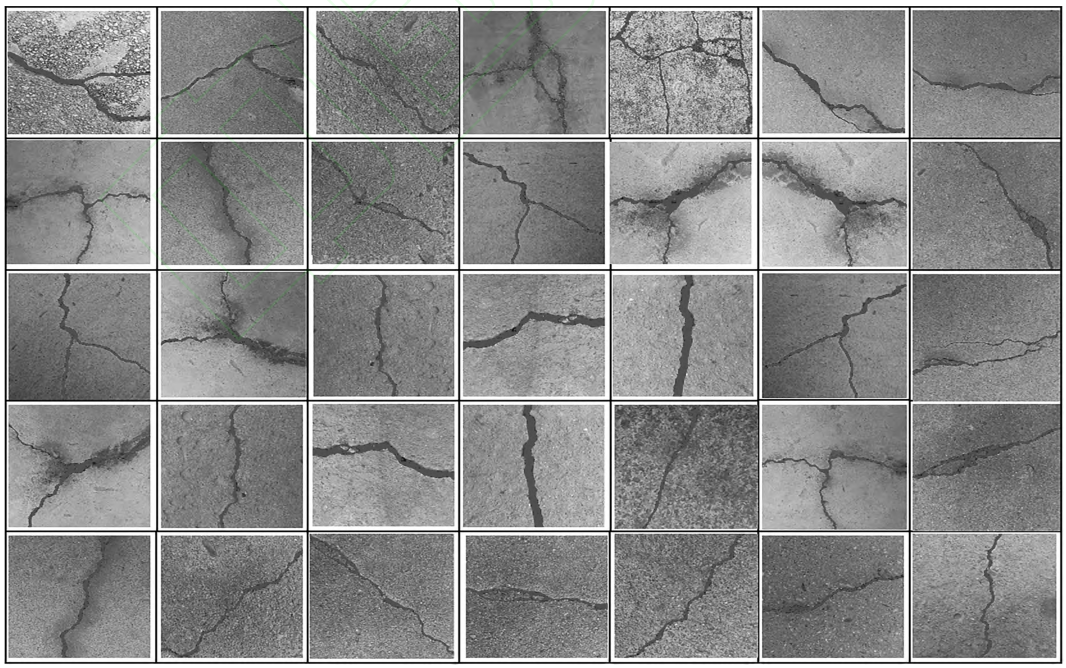


图 17 基于本文算法进行桥梁裂缝检测的部分结果

Fig. 17 Partial results of bridge crack detection based on our algorithm

未来进一步研究的重点是: 在不断提高算法的抗干扰能力和检测准确率的情况下, 进一步提高算法的处理速度, 以便算法在实际的应用过程中, 表现出更好的性能. 针对这一问题, 可以使用 CUDA、MMX、SSE、SSE2 等策略对算法进行优化.

为了推动本文算法的进一步改进和方便其他研究者使用本文算法进行对比和实验, 本文对论文中所使用的桥梁裂缝图像数据集合、DBCC 模型的网络配置文件、超参数配置文件进行开源. 相应的数据集和文件可在如下的链接中得到, 具体的链接为: (https://github.com/maweifei/Bridge_Crack_Image_Data).

References

1 Ma Jian, Sun Shou-Zeng, Yang Qi. *Review on China's bridge engineering research:2014*. China Journal of Highway and Transport, 2014,**27**(5):1−96
(马建, 孙守增, 杨琦. 中国桥梁工程学术研究综述.2014. 中国公路学报, 2014,**27**(5): 1−96)

2 Wei Wu, Wang Jun-Jie, Cai Zhao-Xiong. *Bridge crack detection based on wavelet and radon transform*. Computer Engineering and Design, 2013,**34**(9):3151−3157
(魏武, 王俊杰, 蔡钊雄. 基于小波和 Radon 变换的桥梁裂缝检测. 计算机工程与设计, 2013,**34**(9): 3151−3157)

3 Oh H, Garrick N W, Achenie L E K. *Segmentation algorithm using iterative clipping for processing noisy pavement images*. In:Proceedings of 2th International Conference on Imaging Techmologies: Techniques and Applications in Civil Engineering. Davos, Switzerland. 1998:138−147

4 Li Q Q, Liu X L. *Novel approach to pavement image segmentation based on neighboring difference histogram method*. In:Proceedings of 2008 Congress on Image and Signal Processing. Sanya, Hainan, China:IEEE,2008. 792−796

5 Anders L, Matthew J. *Morphology-based crack detection for steel slabs*. IEEE Journal of selected topics in signal processing, 2012, **6**(7):866−875

6 Varadharajan S, Jose S, Sharma K, Wander L, Mertz C. *Vision for road inspection*. In:Proceedings of 2014 IEEE Winter Conference on Applications of Computer Vision. Steamboat Springs, CO, USA:IEEE, 2014. 115−122

7 Jahanshahi M R, Masri S F, Padgett C W. *An innovative methodology for detection and quantification of cracks through incorporation of depth perception*. Machine Vision and Applications, 2013, **24**(2):227−241

8 Amhaz R, Chambon S, Idier J, Baltazart V. *Automatic crack detection on two-dimensional pavement images: an algorithm based on minimal path selection*. IEEE Transactions on Intelligent Transportation Systems, 2016, **17**(10):2718−2729

9 Zou Q, Cao Y, Li Q, Mao Q, Wang S. *Crack tree: automatic crack detection from pavement images*. Pattern Recognition Letters, 2012, **33**(3):227−238

10 Oliveira H, Correia P L. *Automatic road crack detection and characterization*. IEEE Transactions on Intelligent Transportation Systems, 2013, **14**(1):155−168

11 Tien S N, Begot S, Duculty F, Avila M. *Free-form anisotropy: a new method for crack detection on pavement surface images*. In:Proceedings of 2011 18th IEEE International Conference on Image Processing. Brussels, Belgium:IEEE, 2011. 1069−1072

12 Mustafa R, Mohamed E A. *Concrete crack detection based multi-block CLBP features and SVM classifier*. Journal of Theoretical and Applied Information Technology, 2015, **81**(1):151−160

13 Nguyen H N, Kam T Y, Cheng P Y. *An automatic approach for accurate edge detection of concrete crack utilizing 2D geometric features of crack*. Journal of Signal Processing Systems, 2014, **77**(3):221−240

14 Arena A, Piane C D, Sarout J. *A new computational approach to cracks quantification from 2D image analysis: application to micro-cracks description in rocks*. Computers & Geosciences, 2014, **66**(1):106−120

15 Quintana M, Torres J, Menendez J M. *A simplified computer vision system for road surface inspection and maintenance view document*. IEEE Transactions on Intelligent Transportation Systems, 2016, **17**(3):608−619

16 Shi Y, Cui L, Qi Z, Meng F, Chen Z. *Automatic road crack detection using random structured forests*. IEEE Transactions on Intelligent Transportation Systems, 2016, **17**(12):3434−3445

17 Zhang D, Li Q, Chen Y, Cao M, He L . *An efficient and reliable coarse-to-fine approach for asphalt pavement crack detection*. Image and Vision Computing, 2017, **57**(C):130−146

18 Hassan N, Mathavan S, Kamal K. *Road crack detection using the particle filter*. In:Proceedings of International Conference on Automation and Computing. University of Huddersfield, UK. 2017. 1−6

19 Oliveira H, Correia P L. *Road surface crack detection: improved segmentation with pixel-based refinement*. In:Proceedings of European Signal Processing Conference. Kos, Greece. 2017. 2026−2030

20 Simonyan K, Zisserman A. *Two-stream convolutional networks for action recognition in videos*. Advances in Neural Information Processing Systems, 2014, **1**(4):568−576

21 Zeiler M D, Fergus R. *Visualizing and understanding convolutional networks*. In:Proceedings of European Conference on Computer Vision. Springer, Cham. 2014. 818−833

22 Krizhevsky A, Sutskever I, Hinton G E. *ImageNet classification with deep convolutional neural networks*. In:Proceedings of International Conference on Neural Information Processing Systems. Curran Associates Inc. 2012. 1097−1105

23 Szegedy C,Liu W, Jia Y. *Going deeper with convolutions*. In:Proceedings of Conference on Computer Vision and Pattern Recognition. Boston, MA, USA:IEEE, 2015. 1−9

24 Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z. *Rethinking the inception architecture for computer vision*. In:Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition. Las Vegas, NV, USA:IEEE, 2016. 2818−2826

25 Girshick R. *Fast R-CNN*. In:Proceedings of 2015 IEEE International Conference on Computer Vision. Santiago, Chile:IEEE,2015. 1440−1448

26 Ren S, He K, Girshick R. *Faster R-CNN: towards real-time object detection with region proposal networks*. In:Proceedings of IEEE Transaction on Pattern Analysis and Machine Intelligence. IEEE, 2017.91−99

27 Krizhevsky A, Hinton G. *Convolutional deep belief networks on cifar-10*. [Online], available: http://www.cs.toronto.edu krizindex.html, 2010.

28 Hinton G E, Srivastava N, Krizhevsky A. *Improving neural networks by preventing co-adaptation of feature detectors*. Computer Science, 2012, **3**(4):212−223

29 Lin, M, Chen, Q, Yan, S. *Network in network*. arXiv Preprint arXiv:1312.4400,2014

李良福 陕西师范大学计算机学院副教授. 2006 年获西安交通大学博士学位, 美国卡内基梅隆大学机器人研究所访问学者. 研究方向: 计算机视觉、深度学习、人工智能等.
E-mail: longford@xjtu.edu.cn
(**LI Liang-Fu** Associate professor of college of computer science of Shaanxi Normal University. He received his doctor degree from Xi'an Jiao Tong University in 2006. He was a visiting scholar of robotics institute, Carnegie Mellon University, USA. His research interest covers computer vision, deep learning, artificial intelligence, etc.)

马卫飞 陕西师范大学计算机科学学院硕士研究生. 研究方向: 数字图像处理、计算机视觉、深度学习、人工智能等. 本文通信作者.
E-mail: weifei@snnu.edu.cn
(**Ma Wei-Fei** A graduate student of college of computer science of Shaanxi Normal University.His research interest covers digital image processing, computer vision, deep learning, artificial intelligence,etc.Corresponding author of this paper.)

李 丽 陕西师范大学计算机科学学院硕士研究生, 研究方向: 数字图像处理、计算机视觉、深度学习等.
E-mail: lili94815@outlook.com
(**Li Li** A graduate student of college of computer science of Shaanxi Normal University.Her research interest covers digital image processing,computer vision,deep learning,etc.)

陆 铖 陕西师范大学计算机学院副教授. 2013 年获加拿大阿尔伯塔大学博士学位. 研究方向: 图像处理、模式识别等.
E-mail: chenglu@snnu.edu.cn
(**Lu Cheng** Associate professor of college of computer science of Shaanxi Normal University. He received his doctor degree from University of Alberta in Canada in 2013. His research interest covers image processing, pattern recognition, etc.)