DOI:10.14048/j.issn.1671-2579.2023.02.010

基于YOLO v3深度学习算法的道路裂缝识别模型研究

苏卫国,王景霄

(华南理工大学 土木与交通学院,广东 广州 510000)

摘要:针对道路裂缝检测识别需人工参与、传统算法识别不准确等问题,提出一种基于YOLO v3深度学习算法的道路裂缝识别方法。首先将数据集图片缩放成416×416,然后利用Labelme对数据进行裂缝标注并对边界框位置信息进行转换,最后利用YOLO v3算法框架进行模型训练。结果表明:YOLO v3算法的精确率、召回率、F1分数都大于95%,图片检测速度达到0.123 1 s/张。YOLO v3深度学习算法在精度和速度上都满足了道路裂缝实时检测的要求。

关键词:道路裂缝;深度学习;YOLO v3;边界框

中图分类号:U416.2 **文献标志码**:A

0 前言

根据“十四五”综合交通运输发展主要指标,中国公路总里程在2025年底将会达到550万km,高速公路里程也将会突破19万km。截至2020年底,中国公路总里程已超过535万km,高速公路里程也已达到17.7万km。庞大且复杂的公路交通网使得公路日常养护问题日益突出,由于受公路施工工艺、交通流量、气候条件等因素的影响,公路路面在使用过程中无法避免会产生各种裂缝,进而对道路行车安全产生威胁。为了更好地保障道路使用性能与保证行车安全,必须采用快速且高效的道路裂缝检测方法。

传统的路面裂缝检测识别方法主要是依靠人工进行检测,但人工检测工作效率低,作业风险系数大,且容易受主观因素影响。由于近年来计算机视觉和图像检测、目标识别技术的快速发展,研究者通过人工选取裂缝特征,实现对路面裂缝的自动检测。但人工选取特征具有很强的主观性,在特征选择上的优劣决定了路面裂缝检测的性能,常用特征选择方法包括基于直方图估计、梯度方向直方图、局部二值模式、Gabor滤波和多特征融合等。Saar等[1]利用模板对裂缝边缘特征进行提取,然后利用膨胀处理操作和迭代阈值得到裂缝检测图,在沥青路面裂缝检测方面取得了较好的效果;阮崇武[2]提出了同时利用裂缝特定模板分割、连通域线性、区域信息去噪等图像处理技术提取裂缝特征,大大提高了道路裂缝检测率。虽然这些方法具有较好的检测性能,但是它们对输入图像的质量要求较高,在外界环境发生变化、光线不均匀和有噪声干扰时,裂缝检测的性能将受到影响,鲁棒性较差。

伴随深度学习技术广泛应用于各领域,其在图像和语音的分类与识别等方面取得了巨大的成功,研究者在基于深度学习算法的图像裂缝检测方面进行了大量研究并取得一定成果。Zhang等[3]提出了一种基于CNN模型的裂缝识别算法,并公开了试验所使用的路面图像数据集;Mandal等[4]提出了一种基于YOLO v2深度学习框架的自动道路裂缝检测系统,公开了与研究相关的代码和经过训练的模型;Fan等[5]在采用深度卷积神经网络进行图像裂缝目标识别的基础上,根据自适应阈值法提取目标裂缝的信息;柏嘉洛[6]首先使用一种数据融合方法生成大量所需训练样本,设计了一个解码器-编码器结构的深度卷积神经网络进行端到端的自动道路裂缝检

收稿日期:2023-02-10(修改稿)

基金项目:国家自然科学基金资助项目(编号:51778242)

作者简介:苏卫国,男,博士,副教授.E-mail:suwg@scut.edu.cn

测；冯卉[7]在卷积神经网络模型的基础上，采用不同的卷积核进行多尺度特征图采样，同时使用图像注意力机制给予不同尺度的特征图不同的权重，从而能够较好地识别道路裂缝；Yang等[8]提出了一种基于SSD深度学习框架的自动路面裂缝检测系统，并加入感受野模块来提高网络的特征提取能力。

基于深度学习的图像检测模型可以分为一阶段与两阶段，其中两阶段有Mask-Rcnn和Faster-Rcnn系列，特点是速度慢，效果较好；一阶段有YOLO系列，优点是检测速度快，但效果相对较差。道路裂缝检测工作量大，任务重，需要短时间内完成，YOLO系列算法天生灵活，检测速度快且效果能够基本满足道路裂缝检测的需求。YOLO v3是YOLO系列的第三代算法，在YOLO v1和YOLO v2的基础上进一步改进，YOLO v3检测速度和mAP值都超过了其他算法。本文将运用YOLO v3算法进行道路裂缝检测，并与其他算法进行对比分析，以验证YOLO v3在道路裂缝检测中的良好效果与高效率。

1　YOLO系列算法介绍

YOLO系列是基于深度学习算法的一种回归模型，训练过程中，与R-CNN系列对候选框提取与输出分类结果分两阶段进行不同，YOLO系列直接输出边界框的位置信息与分类结果。YOLO v3算法[9]在前系列（即YOLO v1和YOLO v2）的基础上有所保留和改进，所以首先介绍YOLO v1[10]和YOLO v2[11]。

1.1　YOLO v1

YOLO v1的主要思想是将整幅图像作为深度学习网络的原始输入，然后在最后输出层对边界框的位置信息和类别进行回归预测。第一步是把输入的整张图片划分成$S\times S$个同等大小的网格单元（Grid Cell），然后对每个网格单元都预测B个边界框（Bounding Box），每个边界框一共预测5个数值：x、y、w、h和Confidence Score。x、y、w、h是用来表示边界框的位置与大小信息，都需要进行归一化处理，使其范围变成0～1。Confidence Score表示置信度，如果网格里没有物体即全为背景，则其值等于0；相反，网格里面有物体（本场景中即为裂缝），其值为所预测的边界框和标注的真实框（Ground Truth）的IoU值。另外，每个网格还需预测C个类别概率，表示边界框中的目标分别为每个类别的概率和边界框识别目标的效果。边界框的Confidence Score乘以C个类别概率，得到边界框属于每个类别的置信度大小。最后只需要通过给定阈值，排除置信度低的边界框，对剩下的边界框进行非极大值抑制处理，就可以实现目标检测。

网络结构如图1所示，图中s为步长，主要采用GoogLeNet网络架构，前面的6层卷积层获取图片不同层级的特征，通过最后的2层全连接层可以得到边界框属于每个类别的置信度和边界框的位置大小信息。最后的输出是$S\times S\times(B\times 5+C)$的张量。

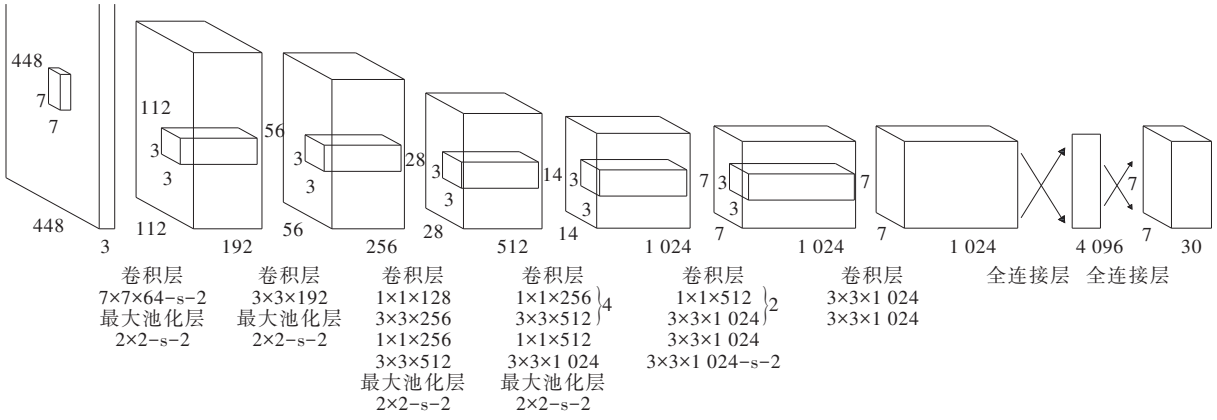

图1　YOLO v1网络架构

YOLO v1作为YOLO系列的第一个版本，其优点很明显，就是整个网络相对简单，能够极大地提升检测效率。缺点是对于距离相对较近的目标，或者尺寸较小的目标，检测效果较差。另外，由于在训练前已经对标注框的大小进行聚类，当实际检测中某一类物体的大小不同于训练样本时，检测效果同样不佳。

1.2 YOLO v2

YOLO v2在v1版本的基础上做了一定的改进，进一步提高了检测的准确率和速度，同时能够识别更多的目标。主要的改进如下：

（1）批量归一化。深度学习过程由于反向传播存在梯度发散的问题，无法更新前面的网络层权重，容易导致深层网络与浅层网络效果相似。批量归一化可以解决这个问题，相当于进行了一定的正则化作用，能够使网络更快更好地收敛。

（2）高分辨率图像分类器。YOLO v1预训练时输入的图像大小为224×224（单位为像素×像素，下同），但在正式训练时使用的图像大小为448×448，由于预训练和正式训练所输入的图像大小不一致，使得模型的检测效果下降。YOLO v2则是首先使用224×224大小的图像进行预训练，然后采用448×448大小的图像对模型权重进行调整，最后才使用448×448的图像进行正式训练，降低了前后分辨率不一致对模型性能的影响。

（3）用先验框。YOLO v1是直接利用最后的全连接层得到边界框位置和大小的预测信息，没有充分利用训练样本中目标的位置与大小信息，导致最终模型的效果不佳。YOLO v2借鉴了先验框的思想，在训练开始前，首先对训练样本中已经标注好的边界框进行聚类，得到了训练样本中目标的边界框空间信息，同时取消了最后的全连接层，在网络的最后直接预测实际边界框的偏移值。

YOLO v2在当时由于其优秀的检测性能备受瞩目，在检测速度上都超过了其他算法，可谓是当时最先进的检测算法，同时由于其可以通过微调网络进行速度与精度的权衡，得到了广泛运用。

1.3 YOLO v3

YOLO v3比YOLO v1和YOLO v2在网络架构上复杂了很多，融合了当下多种流行算法的思想，在小物体检测方面更加准确，另外还可以根据实际情况微调模型结构来获取更快的速度或者更高的精度。改进方面如下：

（1）多尺度预测。一共3个尺度，每个尺度下都可以预测3个边界框，共9个边界框，多于YOLO v2中的5个边界框。尺度1是32倍下采样，特征图大小为13×13；尺度2是对尺度1中的特征图进行上采样（x2），再与16倍下采样的26×26特征图拼接；尺度3同尺度2，是对尺度2中的特征图进行上采样（x2），再与8倍下采样的52×52特征图拼接。多尺度预测如图2所示。

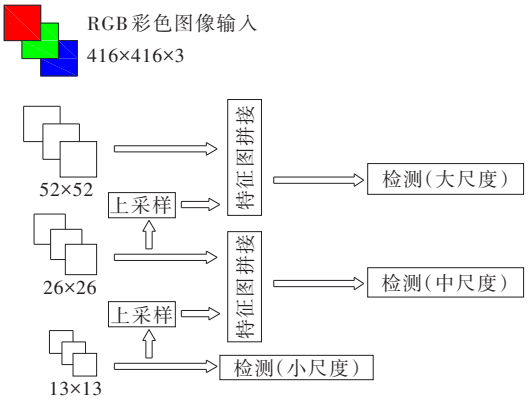

图2 多尺度预测

（2）改进网络结构。在YOLO v2取消全连接层的基础上，进一步取消了池化层，再加入残差连接，它基本融合了当下的经典算法。如图3所示。

	层类型名称	卷积核个数	尺寸/（像素×像素）	特征图大小/（像素×像素）
	卷积层	32	3×3	256×256
	卷积层	64	3×3/2	128×128
1×	卷积层	32	1×1	
	卷积层	64	3×3	
	残差结构层			128×128
	卷积层	128	3×3/2	64×64
2×	卷积层	64	1×1	
	卷积层	128	3×3	
	残差结构层			64×64
	卷积层	256	3×3/2	32×32
8×	卷积层	128	1×1	
	卷积层	256	3×3	
	残差结构层			32×32
	卷积层	512	3×3/2	16×16
8×	卷积层	256	1×1	
	卷积层	512	3×3	
	残差结构层			16×16
	卷积层	1 024	3×3/2	8×8
4×	卷积层	512	1×1	
	卷积层	1 024	3×3	
	残差结构层			8×8
	平均池化层		全局	
	连接层		1 000	
	归一化指数函数			

图3 YOLO v3网络架构

（3）多标签任务预测。YOLO v1和YOLO v2都是采用Softmax预测每个边界框所属类别，这样存在的问题就是每个边界框只能属于一个类别。YOLO v3中，使用独立的多个逻辑回归分类器对边界框类别进行预测，每个边界框可以属于多个类别，实现多标签预测。

2　数据预处理

2.1　数据来源

本文使用的裂缝数据集来源于相关文献的数据集以及自行收集的真实道路裂缝图像，包括CRACK500、GAPs384、TITS、CrackTree200等公开数据集。CRACK500数据集[12]包含500幅裂缝图像，每幅图像分辨率为2 560×1 440；GAPs384数据集[13]包含384幅裂缝图像，每幅图像分辨率为1 920×1 080；TITS数据集[14]包含224幅裂缝图像，每幅图像分辨率为460×320；CrackTree200数据集[15]包含206幅裂缝图像，每幅图像分辨率为800×600；另有自行收集的正式道路裂缝数据集，包含686张裂缝图像，每幅图像分辨率为1 024×1 024。数据集大小一共为2 000张裂缝图片，同时包含了水泥混凝土路面和沥青混凝土路面。不同于大部分的目标检测任务，道路裂缝数据集中不需要负样本图片（即不包含裂缝的路面图像），这是因为道路裂缝图像中绝大部分是背景（即除去裂缝以外的路面部分），这些背景已经是相当大量的负样本。

2.2　图片缩放

由于收集的裂缝图片分辨率大小不一致，为了加快模型收敛速度，所以在训练模型前先将所有图片缩放至原模型的输入分辨率大小416×416。具体方法如下：为保证缩放前后图像比例保持一致，将原图像像素较大的边长按照某一比例缩放至416，剩下的一边按照同样比例缩放，对于缩放后416×416图片的未填充部分进行灰色填充。

2.3　图片标注

训练模型时需要告诉模型目标的位置信息，即在训练前首先要人工手动对数据集中的裂缝进行标注，无论是公开数据集还是自行收集的数据集，都需要将图中的裂缝用边界框标注出来。本研究运用Labelme标注工具进行道路裂缝标注，Labelme是Python下的第三方标注工具包。选用矩形边界框进行标注。本文的裂缝数据集中包括了各种裂缝类型，如横向裂缝、纵向裂缝、块状裂缝、龟裂裂缝，为了简单起见，本研究并未对裂缝进行分类，均标注为一种类型裂缝（Crack）。

2.4　标注框转换

通过Labelme标注得到的数据为边界框位置x_1、x_2、y_1、y_2（相对图片左上角原点）和图片大小img_h、img_w。YOLO v3模型的边界框位置信息为x、y、w、h。x和y表征边界框的中心坐标位置，采用其相对偏移值而非实际坐标值，值域为0～1；w和h表征边界框的宽度与长度，用边界框的实际宽度与长度分别除以图片的宽度与长度，进行归一化处理，值域同样为0～1。x_1、x_2、y_1、y_2、img_h、img_w和x、y、w、h之间的转换关系如式(1)～(4)所示：

$$x=\frac{[(x_1+x_2)/2-1]}{\text{img\_}w} \tag{1}$$

$$y=\frac{[(y_1+y_2)/2-1]}{\text{img\_}h} \tag{2}$$

$$w=\frac{x_2-x_1}{\text{img\_}w} \tag{3}$$

$$h=\frac{y_2-y_1}{\text{img\_}h} \tag{4}$$

转换前的json格式数据如图4所示，转换后的txt格式数据如图5所示，图5中第一个数字0表示标注类别Crack，本文只有一个标注类别，序号即为0。


```
"points":[
[
  100.3612334801762,
  178.85462555066078
],
[
  319.74449339207047,
  233.0396475770925
]
],
"imageHeight":416,
"imageWidth":416
```


图4　转换前的json格式边界框数据

0 0.5012019230769231 0.4915865384615385 0.5264423076923077 0.13221153846153846

图5　转换后的txt格式边界框数据

3　试验结果与分析

3.1　先验框生成

由于YOLO v3算法框架的原始边界框尺寸大小是基于COCO数据集（一共有80个分类）确定的，而本研究只有一个分类（Crack裂缝），且裂缝这一目标和COCO数据集的目标有较大差别，所以原先设置

的边界框尺寸不适合本研究，需要在训练模型前对标注的真实边界框尺寸进行聚类分析。本研究采用K-means算法对训练集中标注好的图像样本进行无监督学习，聚类生成9个先验框尺寸。

聚类分析过程如下：① 首先给定9个聚类中心点，给出9个聚类中心的长度与宽度尺寸；② 计算所有标注框和9个聚类中心点的距离，使标注框与所给出的聚类中心的中心重合，距离 $d=1-IoU$（IoU 为两者之间的交并比）。标注框与聚类中心点的距离越小，说明该标注框与该聚类中心尺寸越接近，将其归于这一聚类中心；③ 首轮聚类结束后，按照已分配好的聚类生成新的聚类中心的长宽尺寸，等于该聚类中宽和高的平均值；④ 循环②、③步骤，直至每个聚类中心的长宽尺寸变化小于阈值。最终得到的先验框如表1所示。

表1 先验框大小

特征图/(像素×像素)	感受野	先验框尺寸/(像素×像素)		
13×13	大	38×54	44×66	48×103
26×26	中	22×46	28×50	29×63
52×52	小	12×24	15×31	22×35

3.2 模型训练

采用YOLO v3算法对图像中的道路裂缝进行识别与检测，基于迁移学习的方法，使用预训练模型作为基础特征提取网络。模型迭代参数epochs设置为101，批处理参数batch_size设置为2，采用darknet53.conv.74作为预处理权重，优化方法为自适应梯度下降法，验证集比例为0.2，其余参数采取模型默认参数。由于训练集较小，最后train loss稳定在0.5左右。本试验的计算机环境配置为Windows 10操作系统、CUDA 10.1.236、CUDNN 7.2.1、Intel(R) Core(TM) i5-7300HQ CPU、Nvidia GeForce GTX 1050Ti显卡，深度学习框架为Pytorch，调用GPU进行加速训练。

3.3 评价指标

使用精确率(Precision)、召回率(Recall)和$F1$分数($F1$-score)作为模型评价指标，具体计算方法见式(5)～(7)。精确率表示在被模型分为正例的样本中，实际为正例的比例，即边界框选出的目标中真实为裂缝所占的比例。召回率则是在实际为正例的样本中，能够被模型准确预测为正例的样本的占比，即图片中所有的裂缝被边界框框出的比例。精确率和召回率是两个矛盾的评价指标，无法同时取得最大，而$F1$分数则综合考虑了精确率和召回率两个评价指标，$F1$越大一般代表该模型效果越好。

精确率P：

$$P=\frac{T_P}{T_P+F_P} \tag{5}$$

召回率R：

$$R=\frac{T_P}{T_P+F_N} \tag{6}$$

$F1$分数：

$$F1=\frac{2PR}{P+R} \tag{7}$$

式中：T_P为真阳性，将正例(裂缝)预测为正例(裂缝)的数量；F_P为假阳性，将负例(非裂缝)预测为正例(裂缝)的数量；F_N为假阴性，将正例(裂缝)预测为负例(非裂缝)的数量。

3.4 试验结果与分析

训练模型结束后，采用CFD数据集作为测试集评价模型。CFD数据集用于文献[16]的试验部分，一共包含了118张裂缝图像，YOLO v3模型检测结果如图6所示。

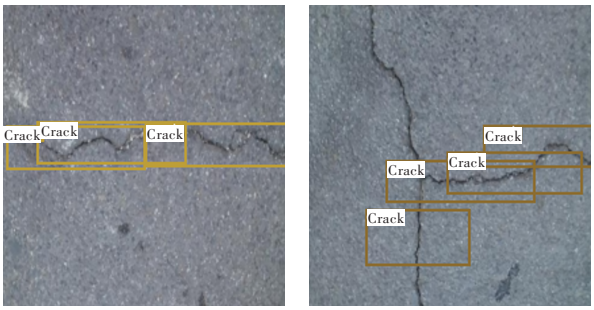

图6 YOLO v3模型检测结果

由图6可知：裂缝能够被识别出来，但也存在少量漏检、错检的情况，这是因为训练集较小以及迭代次数也较小导致的。由于CFD公开数据集被广泛用于其他模型研究的测试中，所以通过横向对比YOLO v3裂缝识别模型与先前研究的相关模型在同一数据集上的效果，可以证明YOLO v3深度学习算法用于道路裂缝识别的优秀性能。本文中提出的模型与其他不同裂缝识别模型在该数据集上的识别结果如表2所示。

表2　CFD公开数据集测试结果

评价模型	精确率	召回率	*F*1分数
YOLO v3	0.954 7	0.958 5	0.956 6
Cross Forest[13]	0.834 3	0.899 5	0.865 7
CrackTree[12]	0.732 2	0.764 5	0.708 0
DenseCrack-Syn-2p[6]	0.978 5	0.981 6	0.980 0

由表2可知:YOLO v3模型在精确率、召回率、*F*1分数等评价指标上远远优于Cross Forest[13]、CrackTree[12]模型,这也说明了深度学习的优越性,在识别能力上远大于传统模型;YOLO v3模型的评价指标略小于DenseCrack-Syn-2p[6],这是由于YOLO v3算法是一阶段的算法,同时本研究训练数据较少(1 510张,远小于DenseCrack-Syn-2p中的8 403张)。通过测试,YOLO v3模型的单张图片检测速度只有0.123 1 s/张,在精度和速度上都满足了道路裂缝检测的要求。

4　总结与展望

提出了基于YOLO v3模型来解决道路裂缝识别这一问题。首先需要对数据进行预处理,包括图片大小缩放、图片标注及转换,然后运用YOLO v3算法进行模型训练,结果表明YOLO v3模型的精确率、召回率和*F*1分数等评价指标都达到了95%以上,速度与精度都满足了道路裂缝识别检测的要求。

但本研究的训练数据相对较少,模型的性能还未达到最佳,可增加训练数据以提升模型的精度。另外,为了简单起见,本研究只有一个类别,并未对裂缝进行分类,今后可以在标注裂缝时进行分类,以更好地实现对裂缝的检测识别。同时并未改动YOLO v3算法的内部架构,后续研究可以考虑微调内部架构使其更符合道路裂缝识别这一目标。

参考文献:

[1] SAAR T, TALVIK O. Automatic asphalt pavement crack detection and classification using neural networks[C]//2010 12th Biennial Baltic Electronics Conference,2010:345-348.

[2] 阮崇武.基于图像的道路裂缝检测与分类算法研究[D].武汉:华中科技大学,2015.

[3] ZHANG L, YANG F, ZHANG Y D, et al. Road crack detection using deep convolutional neural network[C]//2016 IEEE International Conference on Image Processing,2016:3708-3712.

[4] MANDAL V, UONG L, ADU-GYAMFI Y. Automated road crack detection using deep convolutional neural networks [C]//2018 IEEE International Conference on Big Data (Big Data),2019:5212-5215.

[5] FAN R, BOCUS M J, ZHU Y L, et al. Road crack detection using deep convolutional neural network and adaptive thresholding[C]//2019 IEEE Intelligent Vehicles Symposium (IV),2019:474-479.

[6] 柏嘉洛.基于深度学习的端到端道路裂缝检测技术研究[D].武汉:华中科技大学,2019.

[7] 冯卉.基于深度学习的道路裂缝识别算法研究与实现[D].北京:北京邮电大学,2019.

[8] YANG J, FU Q, NIE M X. Road crack detection using deep neural network with receptive field block[J]. IOP Conference Series: Materials Science and Engineering, 2020, 782(2): 042033.

[9] REDMON J, FARHADI A. YOLO v3: An incremental improvement[EB/OL]. 2018: arXiv: 1804.02767. https://arxiv.org/abs/1804.02767

[10] REDMON J, DIVVALA S, GIRSHICK R, et al. You only look once: Unified, real-time object detection[C]//2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR),2016:779-788.

[11] REDMON J, FARHADI A. YOLO9000: better, faster, stronger [C]//2017 IEEE Conference on Computer Vision and Pattern Recognition,2017:6517-6525.

[12] ZHANG L, YANG F, ZHANG Y D, et al. Road crack detection using deep convolutional neural network[C]//2016 IEEE International Conference on Image Processing (ICIP), 2016:3708-3712.

[13] EISENBACH M, STRICKER R, SEICHTER D, et al. How to get pavement distress detection ready for deep learning? A systematic approach[C]//2017 International Joint Conference on Neural Networks (IJCNN),2017:2039-2047.

[14] AMHAZ R, CHAMBON S, IDIER J, et al. Automatic crack detection on two-dimensional pavement images: An algorithm based on minimal path selection[J]. IEEE Transactions on Intelligent Transportation Systems,2016,17 (10):2718-2729.

[15] ZOU Q, CAO Y, LI Q, et al. CrackTree: Automatic crack detection from pavement images[J]. Pattern Recognition Letters,2012,33(3): 227-238.

[16] SHI Y, CUI L M, QI Z Q, et al. Automatic road crack detection using random structured forests[J]. IEEE Transactions on Intelligent Transportation Systems,2016,17 (12):3434-3445.