An Efficient Method for Detecting Asphalt Pavement Cracks and Sealed Cracks Based on a Deep Data-Driven Model

Nan Yang, Yongshang Li and Ronggui Ma *

School of Information Engineering, Chang’an University, Xi’an 710064, China
* Correspondence: rgma@chd.edu.cn; Tel.: +86-186-9188-7916

Abstract: Thanks to the development of deep learning, the use of data-driven methods to detect pavement distresses has become an active research field. This research makes four contributions to address the problem of efficiently detecting cracks and sealed cracks in asphalt pavements. First, a dataset of pavement cracks and sealed cracks is created, which consists of 10,400 images obtained by a vehicle equipped with a highway condition monitor, with 202,840 labeled distress instances included in these pavement images. Second, we develop a dense and redundant crack annotation method based on the characteristics of the crack images. Compared with traditional annotation, the method we propose generates more object instances, and the localization is more accurate. Next, to achieve efficient crack detection, a semi-automatic crack annotation method is proposed, which reduces the working time by 80% compared with fully manual annotation. Finally, comparative experiments are conducted on our dataset using 13 currently prevailing object detection algorithms. The results show that dense and redundant annotation is effective; moreover, cracks and sealed cracks can be efficiently and accurately detected using the YOLOv5 series model and YOLOv5s is the most balanced model with an F1-score of 86.79% and an inference time of 14.8ms. The pavement crack and sealed crack dataset created in this study is publicly available.

Keywords: asphalt pavement; crack detection; semi-automatic crack annotation; data-driven model; deep learning

1. Introduction

As urbanization increases, highway networks have been developed to meet challenges and demands in transportation. As highways comprise part of the critical infrastructure for public transportation, promoting advanced analysis and assessment technology for highway systems is an important part of an intelligent transportation system (ITS). In the past decade, an increasing number of highways have been damaged due to the poor environment, car overload, material aging, and so on [1]. Pavement distress detection, identification and classification are important steps in a pavement management system (PMS). It helps the agency to determine the appropriate rehabilitation techniques to be performed on the pavement [2]. Generally, cracks are the earliest signs of pavement distress. The continuous propagation of cracks without proper treatment in the early stages of damage will result in high maintenance costs and severe consequences. Therefore, detecting cracks and repairing them quickly are essential tasks for highway maintenance departments. Normally, highway maintenance workers use sealants to repair cracks. However, when the pavement structure has been damaged, cracks are often more likely to develop around the sealed crack, so it is also necessary to detect sealed cracks. The traditional routine of highway pavement distress inspection relies on manual on-site surveys, which are labor-intensive and time-consuming. In addition, highways are dangerous working environments for road inspection personnel [3]. It is therefore necessary to develop automatic and efficient methods of detecting highway pavement cracks and sealed cracks.
At present, special road condition inspection devices have been extensively studied. Non-contact devices such as thermal imaging sensors [4] and ground-penetrating radar [5] have been utilized to detect cracks and take advantage of the differences in the signals returned from normal and damaged pavements. Embedded fiber optic sensors [6] are also an emerging technology used to detect pavement distress. Although the tools mentioned above can accurately locate pavement distress, their high cost and low efficiency are critical drawbacks in real-world scenarios, preventing them from being applied to a wider range of situations. Image acquisition methods based on charge-coupled devices (CCDs) and complementary metal oxide semiconductor (CMOS) sensors are prevalent because of their efficiency and low cost. However, the detection of pavement distress using image processing is still a very challenging task, especially for asphalt pavements. The reasons for this can be summarized as follows:

- The irregularity of the pavement surface [3];
- Cracks are characterized by low-level features that can be easily confused with the background texture and other objects on the pavement [7];
- The influence of external conditions such as illumination and camera jitter on the image quality.

Image processing methods such as transformation, enhancement, and segmentation have gained the attention of researchers in the field of pavement assessment, but these methods are empirical [3], i.e., they require constant adjustment of the parameters to achieve an optimum result. Traditional machine learning methods, such as random forest [8,9] and AdaBoost [10], have also been used to detect cracks in pavements. The problem is that such methods can only obtain low-level image information and cannot extract high-level semantic information, which has a significant impact on the robustness of the algorithms.

With the increase of parallel computing power and the development of deep learning, data-driven methods are widely used in PMS in various countries [11,12]. Convolutional neural network (CNN) [13] is one of the well-known data-driven methods that can automatically learn high-level information from large amounts of data through a multilayered artificial neural network (ANN), which can be used to classify images or detect objects.

Nevertheless, challenges still exist for data-driven-based pavement crack detection:

- Data are the basis of deep learning algorithms, but there are not enough publicly available pavement datasets, and even models trained on publicly available datasets with only a few hundred images are not guaranteed to be effective.
- Building datasets is a time- and resource-consuming task.
- There is no efficient and practical detection method for pavement cracks and sealed cracks.

The crack detection system proposed in this study was designed to address these problems. In a real-world scenario, the low practicality and inefficiency of pavement inspection systems are difficult problems for road maintenance departments. Research institutions and companies in many countries have developed automated road condition monitoring vehicles that automatically detect road damage at normal traffic flow speeds while a large number of road images are collected using CCD or CMOS sensors mounted on the rear of the vehicle. The use of an efficient and practical system to process this enormous amount of data and detect cracks and sealed cracks within them was the focus of this study. The contributions of this study are as follows:

1. A publicly available dataset of pavement cracks and sealed cracks, collected by an automated road condition monitoring vehicle, is created and labeled;
2. For the features of highway asphalt pavement images, a dense and redundant crack annotation method is proposed, which provides more object instances and more accurate object positioning than traditional big-block annotation;
3. In order to quickly implement the inspection system and reduce labor costs, a semi-automatic crack annotation method is proposed, which reduces the creation time of the dataset by 80% compared with fully manual annotation;
4. In our dataset, 13 currently popular object detection models are compared, and the YOLOv5 family of models are considered to be efficient and accurate. The results prove that our proposed annotation method is effective.

The paper’s structure comprises five sections, complemented by the current Introduction. Section 2 presents related work in the field of crack detection in recent years. Details of our newly developed dataset are presented in Section 3. Section 4 describes the specific experimental settings. Section 5 illustrates the results, including a detailed comparison of different models and a discussion. Section 6 concludes the study.

2. Related Work

In this section, we present various studies that were proposed by different researchers for pavement distress detection. As shown in Table 1, we focus on the pavement distress dataset, image annotation type, and detection model.

Table 1. Recent studies on crack detection (N/A—Not Available).

<table>
<thead>
<tr>
<th>Image Dataset</th>
<th>Image Annotation Type</th>
<th>Detection Model</th>
<th>Researchers/Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>2468 grayscale images with 6 distress types, collected by a measurement vehicle</td>
<td>Patch-level</td>
<td>LeNet-based model for crack classification</td>
<td>Eisenbach et al. [14] 2017</td>
</tr>
<tr>
<td>48,000 crack or non-crack images, collected from Google Street View</td>
<td>Patch-level</td>
<td>VGG-based model for crack classification</td>
<td>Maniat et al. [15] 2021</td>
</tr>
<tr>
<td>400 crack images from routine civil construction</td>
<td>Object-level</td>
<td>Faster R-CNN model for crack detection</td>
<td>Wang et al. [16] 2020</td>
</tr>
<tr>
<td>903 images with 8 different distress types, captured by a smartphone</td>
<td>Object-level</td>
<td>SSD-based model for crack detection</td>
<td>Maeda et al. [17] 2018</td>
</tr>
<tr>
<td>3100 real-world images captured by digital camera</td>
<td>Object-level</td>
<td>YOLOv2-based model for crack detection</td>
<td>Deng et al. [18] 2021</td>
</tr>
<tr>
<td>280 highway pavement and urban road images</td>
<td>Object-level</td>
<td>SSD-based model for crack detection</td>
<td>Han et al. [19] 2021</td>
</tr>
<tr>
<td>2000 pothole images captured by a smartphone</td>
<td>N/A</td>
<td>GAN-based model for data augmentation</td>
<td>Maeda et al. [20] 2021</td>
</tr>
<tr>
<td>95,000 road crack images</td>
<td>Pixel-level</td>
<td>FCN-based model for crack segmentation</td>
<td>Wu et al. [21] 2021</td>
</tr>
<tr>
<td>Combination of publicly available crack dataset</td>
<td>Pixel-level</td>
<td>DFANet-based [22] model for crack segmentation</td>
<td>Qiao et al. [23] 2021</td>
</tr>
<tr>
<td>Combination of publicly available crack dataset</td>
<td>Pixel-level</td>
<td>Encoder-decoder-based model for crack segmentation</td>
<td>Nguyen et al. [24] 2021</td>
</tr>
<tr>
<td>Combination of publicly available crack dataset</td>
<td>Pixel-level</td>
<td>Encoder-decoder-based model for crack segmentation</td>
<td>Xiang et al. [25] 2020</td>
</tr>
</tbody>
</table>

2.1. Image Dataset of Pavement Distress

Having sufficient data is the cornerstone of using deep learning to detect pavement distress. There have been several attempts by researchers to build pavement image datasets through a variety of image acquisition methods. The pavement damage dataset built by Maeda et al. [17] using a smartphone contains 9053 images with eight different damage types. Here, the smartphone was mounted on top of a vehicle dashboard to capture the road ahead, so the quality of the images was affected by vehicle vibration; moreover, since the images were captured approximately 10 m ahead, only the easily distinguishable road damage, such as potholes, coarse cracks, and blurred white lines, can be found in the images. The German asphalt pavement distress (GAPs) dataset [14] was acquired using a high-resolution camera suspended behind a high-speed measurement vehicle and contains 2468 grayscale images with six damage types, and the original images were segmented into sub-images with a resolution of 64 × 64 pixels for classification. The data in the GAPs dataset come from three different federal roads in Germany, with the data from two of the roads serving as the training and validation sets, and the data from the other road serving as the test set only. The GAPs classification results show that when there is a difference between the training and the test set, the performance of the ANN on the test set deteriorates significantly, even if the ANN has achieved good results on the training
set. The dataset created by Maniat et al. [15] is from Google Street View and the main drawbacks are the presence of many artifacts and the low quality of the images, but using Google Street View is a very economical way to obtain pavement data. Wu et al. [21] used a camera to capture 95,000 images for segmenting cracks, and the method they used is based on a data-driven full convolutional network (FCN), which means that in the training set, each object needs to be accurately labeled manually using polygons, which can be labor- and resource-intensive. It is also a challenge to apply this rapidly in practice, and the data are not publicly available.

Many researchers do not have enough data. Amhaz et al. [26], Oliveira et al. [27], and Shi et al. [8] used a segmentation-based method to detect cracks in pavements, but the number of images in the datasets they used was less than 500, which is not enough to train an ANN to discriminate cracks. Wang et al. [16] used a dataset containing only 20 images to train a crack detector, increasing the diversity of the data by means of data augmentation. Since, in practice, there are not enough pothole images to train an ANN, Maeda et al. [20] proposed the use of a generative adversarial network (GAN) to generate the required images, which are pseudo-images that are difficult to distinguish from real images.

According to our survey, the number of publicly available pavement datasets is not as large as that of natural scenes, and the amount of data is much smaller than what is available in well-known natural scene datasets, e.g., ImageNet [28] and KITTI [29]. There is currently no unified standard for pavement datasets, and many researchers are constrained by the fact that they do not have enough pavement data to support their experiments. Some researchers [23–25] have mixed multiple publicly available datasets or self-collected data to improve the performance of data-driven models, which suggests that publicly available datasets will help drive future developments in pavement damage detection.

In the last few years, road condition monitors have been applied in several countries so that uniform, balanced, and high-quality pavement images can be collected quickly, as these devices are equipped with high-speed cameras and comprehensive illumination systems. In this study, we developed a pavement image dataset of 10,400 images collected by an automatic monitoring vehicle on a section of a highway in Inner Mongolia, China; in addition, we have made the dataset publicly available for future research.

2.2. Image Annotation

Supervised learning is a type of machine learning in which an ANN learns a pattern from the training data and infers new instances based on that pattern. The currently popular data-driven methods for pavement damage detection are usually based on supervised learning, meaning that all objects to be detected in the training data need to be labeled. As mentioned in PASCAL VOC [30] and COCO [31], classification, detection, and segmentation are three different tracks and also correspond to three different data labeling methods. The annotation for the image classification task is at image-level and requires a binary label indicating whether or not the object appears in the image. For pavement damage classification, a common approach is to segment a large image into sub-images, which is also known as patch-level classification [14,32]. The annotation of the object detection task is at object-level, where both the category of the object and its position in the image need to be stated, and the position of the object is usually represented using a bounding box [17–19]. The annotation for the segmentation task is at pixel-level, where each pixel in the image is annotated with the category or the background to which it belongs [23,25].

Patch-level classification and object-level detection yield localization information on pavement damage, but the results are coarse and cannot be used for pavement assessments, whereas image segmentation can generate accurate crack localization and structures, which can be used to classify crack categories and evaluate important crack characteristics such as length and width [33]. In practical applications, we can label crack images at image-level and object-level relatively easily; however, the workload of pixel-level labeling is several times higher than that of the former two [28]. Therefore, a trade-off needs to be made between fine-grained crack detection and the workload of labeling the data.
Considering the realistic practicality, in this study, we propose a redundant and dense object-level annotation style. To improve the efficiency of image annotation, we propose a semi-automatic annotation method (the details are presented in Section 3.2).

2.3. Detection Model

As mentioned before, there are three forms of data annotation for cracks, namely, image-level, object-level, and pixel-level annotation, so there are three corresponding data-processing models. The popularity of classification networks started with AlexNet [34], after which more powerful and efficient ANNs were proposed, such as GoogLeNet [35], ResNet [36], and MobileNet [37]. Their structure usually comprises a series of convolutional blocks used to extract image information and a fully connected layer that follows the convolutional layer for classification. Image classification is a fundamental task in the field of computer vision, and image classification networks are also used as the backbones of object detection and segmentation models, which means that the backbone network affects the results of detection or segmentation. Object detection models are often classified into two categories: one-stage and two-stage. R-CNN [38], Fast R-CNN [39], and Faster R-CNN [40] are two-stage object detection networks. In the first stage, the candidate regions need to be obtained, and then the candidate regions are detected and classified in the second stage. YOLO [41], SSD [42], RetinaNet [43], and FCOS [44] are single-stage target detection models that use a single CNN to detect objects and classify them, so the advantage of these networks is fast processing. The main segmentation models are FCN [45], DeepLab [46], and U-Net [47]. Although these types of networks can generate pixel-level predictions for cracks, we did not use them in this study because pixel-level annotation is too resource- and time-consuming to be applied in realistic scenarios.

In this study, 13 popular object detection models were divided into four groups according to model complexity, and extensive experiments were conducted on the dataset presented here, as shown in Section 4.

3. Proposed Dataset

3.1. Image Acquisition

At present, in the field of pavement distress detection, some researchers have acquired images with cameras or smartphones [17,48]. Although this approach allows one to acquire pavement images of different environments, with different illumination levels and shooting angles to improve the diversity of the data, it is quite inefficient. Unmanned aerial vehicles (UAVs) equipped with HD cameras have also been used to collect pavement images [49,50]. In order to obtain clear images of the road surface, the flight altitude and speed of UAVs are limited, so a suitable scenario for their application is an urban street with a low traffic volume. An automatic road measurement vehicle can collect road images at normal traffic flow speeds, and the high-resolution CCD sensor and LED illumination system ensure the uniformity of the captured images, so these have been applied by road authorities in several countries [3].

The images used in this study were obtained by a road measurement vehicle and collected from the Hulunbuir section of the Suiman Expressway in the Inner Mongolia Autonomous Region, China, as shown in Figure 1. A line scan industrial camera was mounted on the rear of the vehicle. The camera has a CCD sensor resolution of 3024 pixels and scans a road width of 3 m, with 1 pixel representing approximately 1 mm of road. A rotary encoder mounted on the rear axle of the vehicle generates pulses as the wheels move, and this pulse signal triggers the line scan camera to take a picture of the road surface for every 1 mm of vehicle movement. Therefore, 1 mm² of the road surface area corresponds to approximately 1 pixel in the image. In the process of road image collection, the vehicle’s driving speed is between 80 km/h and 120 km/h, and the camera can photograph the road surface uniformly at different vehicle speeds. In order to ensure that the images captured in different environments are balanced and uniform, a powerful integrated LED set is used.
to provide stable illumination conditions, as shown in Figure 2. Examples of the images are shown in Figure 3.

Figure 1. Location and vehicle of data collection: (a) location of the pavement images; (b) the road measurement vehicle; (c) front view of the road measurement vehicle.

Figure 2. CCD and illumination system.

Figure 3. Examples of images collected by the road measurement vehicle: (a) intact road surface; (b) crack image; (c) sealed crack image; (d) crack and sealed crack image.
In this study, we collected 106,792 images with a resolution of 3024 \times 1889 on a 15 km length of road. As mentioned in [17,51], we decided to crop the original images to a resolution of 600 \times 600 pixels, meaning that each image represents a pavement area of approximately 0.36 m².

3.2. Image Annotation

As mentioned regarding ImageNet [28], there are two basic requirements in a bounding box annotation system: quality and coverage. Quality means that each bounding box needs to be tight, i.e., all visible parts of the object must be contained by a minimal bounding box. Coverage means that every instance of an object needs to have a bounding box to tell the algorithm which parts of the image are to be focused on. For common objects, individual instances such as people, chairs, and cars are easy to annotate using a properly scaled bounding box, and the annotated results are generally not ambiguous. However, cracks are different because they do not have a particular structure. For example, Chinese standards for evaluating the performance of highways classify cracks as alligator, block, transverse, or longitudinal cracks. Using only a single tight bounding box to label a transverse or longitudinal crack means that this bounding box will have an incongruous aspect ratio, as shown in Figure 4a. In addition, the study of ImageNet [28] showed that objects with thin structures have the worst localization accuracy. A huge bounding box will appear in an image when the trend of a crack is sloping, and in which the crack occupies only a tiny portion, while most of the pixels represent the background, as shown in Figure 4b. This type of annotation can lead to mismatched labels when training the network [52]. For common items, an object is one entity; however, a whole crack can be considered to be composed of many sub-cracks. Therefore, in this study, we propose a dense and redundant crack labeling method, in which a crack is densely contained by multiple tight small bounding boxes instead of a very long or large bounding box, and the adjacent bounding boxes need to overlap (see Figure 4). We propose this for the following reasons:

- Dense annotations, although still blocky, can show more accurate crack localization and structures relative to patch-level annotations;
- Redundant annotations increase the number of instances in the dataset.

![Figure 4](image_url)

**Figure 4.** A comparison of traditional annotation and our proposed method: (a) a longitudinal crack is labeled with traditional annotation (green) and our dense and redundant annotations (red); (b) a sloping sealed crack is labeled with traditional annotation (green) and our dense and redundant annotations (yellow).

3.2.1. Manual Annotation

CVAT [53] is a free online interactive tool for labeling videos or images. Because it supports multiple formats and allows good image and labeling management, we used it to label the pavement images based on its collaborative mechanism. In this study, cracks and sealed cracks were the objects to be labeled; however, in many current studies, sealed...
cracks are not the main detection targets. The reasons why we labeled the sealed cracks are as follows:

- Road maintenance workers usually use sealant to repair cracks, but since the asphalt road structure has been damaged, cracks will soon reappear around the sealed cracks.
- According to the Chinese highway performance assessment standards [54] and pavement distress identification manual issued by the U.S. Federal Highway Administration [55], for asphalt pavements, sealed cracks are also a type of pavement distress and are used to evaluate the highway maintenance quality.
- Cracks and sealed cracks are treated differently in PMS [56,57].
- According to our observations, sealed cracks, similar to cracks, are the dominant damage class for asphalt highways.

The two key challenges in labeling crack images are consistent labeling standards and the need for all cracks and sealed cracks to be labeled. Our approach involved teams of three people being trained in a standardized lesson before they started formal labeling. Once the labeling was complete, they needed to check each other’s work. That is, an image was labeled by one annotator and checked by two inspectors to ensure that there were no missed objects or incorrect labels.

The number of 600 × 600 pixel sub-images was about 500,000, which is an astronomical amount for a team of three people, and it was almost impossible to annotate them all by hand. We therefore developed a semi-automatic method for crack annotation, described in the next subsubsection.

### 3.2.2. Semi-Automatic Annotation

As mentioned in COCO [31], the annotation of all the data took thousands of worker-hours and was an extremely time-consuming task. We implemented a semi-automatic method consisting of six steps.

- Step 1: We need to manually label some data to train the initial model. One question is how much data we need at a minimum to train a model to provide largely satisfactory results. For image classification tasks, Arya et al. [12] argued that at least 5000 labeled images per category are needed. For object detection, Maeda et al. [17] suggested that at least 1000 images per category are needed, while according to Shahinfar et al. [58], the rate of improvement in model performance starts to level off when the number of images is greater than 150–500. In this study, we manually labeled 800 images as the training data for training the initial model, considering that the annotation method used here generates more instances of each image.

- Step 2: An initial model is trained on the manually labeled dataset in Step 1. The performance of this model will not be particularly good because of the problem of insufficient data, but it is still necessary to make the model relatively optimal by adjusting the hyperparameters. The model we used was YOLOv5s [59].

- Step 3: The trained model is used to detect the unlabeled data and save the results.

- Step 4: The results are reviewed, and manual corrections are made in the case of inaccurate results, including missed instances, mislabeling, and labeling that is not accurate enough.

- Step 5: The corrected data and the initial data are merged into a new training set, and a new model is trained on this dataset.

- Step 6: Steps 3–5 are repeated to continuously update the model until the performance of the model no longer improves significantly.

### 3.3. Data Statistics

The initial 800 images in the dataset were manually annotated by a three-person team, and the annotation and review processes took 8 and 4 worker-hours, respectively, with an average time of 0.9 min per image. In the semi-automatic labeling stage, 800 unlabeled images were fed into the model trained in the previous stage in each cycle. After 12 cycles, 9600 semi-automatically labeled images were obtained in total. In each cycle, we trained a
Step 1: We need to manually label some data to train the initial model. One question is how much data we need at a minimum to train a model to provide largely satisfactory results. For image classification tasks, Arya et al. [12] argued that at least 5000 labeled images per category are needed. For object detection, Maeda et al. [17] suggested that at least 1000 images per category are needed, while according to Shahin-far et al. [58], the rate of improvement in model performance starts to level off when the number of images is greater than 150–500. In this study, we manually labeled 800 images as the training data for training the initial model, considering that the annotation method used here generates more instances of each image.

Step 2: An initial model is trained on the manually labeled dataset in Step 1. The model we used was YOLOv5s [59]. We obtained a total of 13 crack detection models in the manual labeling and semi-automatic labeling stages, and the mAP of each model is shown in Figure 5. At the beginning, the mAP value increased rapidly; after that, the performance of the model barely improved.

Figure 5. The mAP value for each model over 13 cycles.

In the process of reviewing the semi-automatic labeled images, the most annotation errors we found occurred on the pavement’s white lines. White line damage was mistakenly identified by the model as cracks or sealed cracks. Another annotation error was that the model sometimes misidentified pavement stains as cracks or sealed cracks, but this error gradually decreased with the number of training cycles, and a richer dataset would also help to reduce these issues.

Overall, the dataset we developed contains 10,400 pavement images. Among these images, 202,840 bounding boxes were created, of which 132,012 are bounding boxes for cracks and the rest are for sealed cracks. There was a difference between the manually annotated data and the semi-automated annotated data, e.g., in our statistics, the average number of instances per sample was 13 in the first 800 images, but in the other semi-automated annotated image data, the number was 20. The reason for this is our proposed dense and redundant annotation method. Common sense considers a crack to be a complete crack, while we divided each whole crack into sub-cracks, which is the way we labeled them. Each part of the crack has a chance to be detected by the model, thus generating more detected instances than manual annotation. Although the semi-automatic annotation method is more dense and more redundant, after checking, we found that the result was still good because it satisfied the two criteria of crack annotation: consistency and coverage. Figure 6 shows a comparison of manual annotation with semi-automatic annotation.
Figure 6. Comparison of manual annotation and semi-automatic annotation: (a) our dense and redundant manual annotation; (b) semi-automatic annotation, which was denser and more redundant.

4. Experimental Setup

In previous studies on crack detection, we found that lightweight models were often used for subsequent deployment on edge devices [17,60]. It is generally agreed that deeper models have better feature extraction capabilities, but this also means more parameters and a longer training time. Therefore, after a comprehensive consideration, 13 currently prevailing object detection models were used for experiments on the dataset we developed in this study. According to the number of parameters in the models, they were divided into four groups, so that the number of parameters in each group was similar, as shown in Table 2. All these models are open-source. YOLOv5s was the model used in this study to generate pseudo-labels, and we used it as a benchmark in the experiments.

Table 2. The 13 models divided into four groups.

<table>
<thead>
<tr>
<th>Group</th>
<th>Model</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group 1</td>
<td>SSDlite320 MobileNetV3-Large</td>
<td>3.4 M</td>
</tr>
<tr>
<td></td>
<td>YOLOv5n</td>
<td>1.9 M</td>
</tr>
<tr>
<td></td>
<td>YOLOv5s</td>
<td>7.2 M</td>
</tr>
<tr>
<td>Group 2</td>
<td>Faster R-CNN MobileNetV3-Large-FPN</td>
<td>19.4 M</td>
</tr>
<tr>
<td></td>
<td>Faster R-CNN MobileNetV3-Large-320-FPN</td>
<td>19.4 M</td>
</tr>
<tr>
<td></td>
<td>YOLOv5m</td>
<td>21.2 M</td>
</tr>
<tr>
<td>Group 3</td>
<td>FCOS ResNet50-FPN</td>
<td>32.3 M</td>
</tr>
<tr>
<td></td>
<td>RetinaNet ResNet50-FPN</td>
<td>34.0 M</td>
</tr>
<tr>
<td></td>
<td>RetinaNet ResNet50-FPN v2</td>
<td>38.2 M</td>
</tr>
<tr>
<td></td>
<td>SSD300 VGG16</td>
<td>35.6 M</td>
</tr>
<tr>
<td>Group 4</td>
<td>Faster R-CNN ResNet50-FPN</td>
<td>41.8 M</td>
</tr>
<tr>
<td></td>
<td>Faster R-CNN ResNet50-FPN v2</td>
<td>43.7 M</td>
</tr>
<tr>
<td></td>
<td>YOLOv5l</td>
<td>46.5 M</td>
</tr>
</tbody>
</table>

In our experiments, all these models were based on the PyTorch framework. We used a cloud server as the training platform, and the GPU used was an NVIDIA RTX A5000 with 24 GB memory. We adjusted the batch size to the scale of each model to maximize GPU utilization. All these models were pre-trained on COCO [31] or ImageNet [28], but instead of using the pre-trained weights, we trained them from scratch on our dataset.

Precision (Equation 1), recall (Equation 2), F-score (Equation 3), and mAP are common metrics used to evaluate the performance of object detection models. Precision is the proportion of relevant instances in the retrieved instances, while recall is the proportion of relevant instances retrieved, and they are both based on relevance. In Equations (1) and (2), true positive (TP) indicates the correct detection of ground truth, false negative (FN) indicates objects that were not detected, and false positive (FP) indicates incorrect detection. In the post-processing stage, the detection of an object was evaluated by the intersection...
over union (IoU), which indicates the degree of overlap between the predicted result and the true annotation. In this study, we considered IoU > 0.5 to validate the detected instances. The confidence score is also a free parameter and indicates a model’s certainty about the detection results. In this study, all detected instances with confidence scores less than 0.25 were filtered out. The F-score (Equation 3) is a method of combining the precision and recall of the model, and it is defined as the harmonic mean of these. We can adjust $\beta$ (Equation 3) to give more importance to precision over recall, or vice versa. Common adjusted F-scores include the F0.5-score and the F2-score, as well as the standard F1-score. In this study, only the F1-score was considered, which effectively reflects the model’s overall capability. Among the metrics defined for the Pascal VOC challenge [30], mAP was calculated over a single IoU value, 0.5, while COCO [31] was different and more rigorous, in that 10 different IoU thresholds were considered. Both evaluation criteria were considered in the study.

$$\text{Precision} = \frac{TP}{TP + FP}$$  \hspace{1cm} (1)

$$\text{Recall} = \frac{TP}{TP + FN}$$  \hspace{1cm} (2)

$$F_\beta = \frac{\left(\beta^2 + 1\right) \cdot \text{Precision} \cdot \text{Recall}}{\beta^2 \cdot \text{Precision} + \text{Recall}}$$  \hspace{1cm} (3)

Multiply-accumulate operations (MACs) and parameters were used to measure the computational complexity of the ANNs, and these were calculated in the experiments. Training time and inference time are also important factors to be considered in practical applications, so these were recorded and compared. In this study, the training time was measured for every epoch with an NVIDIA RTX A5000, and the inference time was measured for every image with an NVIDIA RTX 2070 SUPER. The inference time included both the pre-processing and post-processing steps. Some images that never appeared in the training and validation sets were used to evaluate the performance of all the models, comprising data from the same measurement vehicle and the same highway.

5. Results and Discussion

In this section, we describe the evaluation of all the models mentioned in Table 2 on the dataset we developed. In this study, 9600 images were randomly selected as the training set, and the other 800 images were used as the validation set. As mentioned in Section 4, all the models were divided into four groups depending on their parameters, and models within the same group were used for comparative studies. We chose this method because the application scenarios of the models with different numbers of parameters are different. Lightweight models can be deployed on devices with lower computing power such as smartphones, while large models require the support of GPUs with higher computing power.

The models in Group 1 are fairly lightweight models, and their results are listed in Table 3. SSDlite320 MobileNetV3-Large is a model based on SSD and MobileNet, and the input images needed to be resized to 320 $\times$ 320 pixels. Its MAC value was much lower than that of YOLOv5n and YOLOv5s, which means that its computational complexity was very low. Except for the computational complexity, this model lagged behind for all other metrics across the board. It is also interesting that despite having the lowest computational complexity, the training time and inference time of SSDlite320 MobileNetV3-Large were longer than those of the other two models. From the confusion matrix (see Figure 7), 4476 detected instances containing only the background were mistakenly considered as cracks and sealed cracks by SSDlite320 MobileNetV3-Large, but for YOLOv5n and YOLOv5s, the numbers of mistaken backgrounds were 3399 and 2771, respectively. In Table 3, we can see that the results of YOLOv5n and YOLOv5s are very close, but the computational complexity of the former is only about a quarter of the latter. Their PR curves and confusion matrices are shown in Figure 7.
Table 3. The results of the models in Group 1.

<table>
<thead>
<tr>
<th>Model</th>
<th>Training Time</th>
<th>Inference Time</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
<th>mAP (COCO)</th>
<th>mAP (Pascal VOC)</th>
<th>MACs</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSDlite320 MobileNetV3-Large</td>
<td>230 s</td>
<td>40.2 ms</td>
<td>65.81%</td>
<td>56.07%</td>
<td>60.55%</td>
<td>14.62%</td>
<td>45.57%</td>
<td>528.96 M</td>
</tr>
<tr>
<td>YOLOv5n</td>
<td>51 s</td>
<td>14.8 ms</td>
<td>80.18%</td>
<td>89.50%</td>
<td>86.79%</td>
<td>55.80%</td>
<td>88.04%</td>
<td>8.218 G</td>
</tr>
<tr>
<td>YOLOv5s</td>
<td>52 s</td>
<td>14.8 ms</td>
<td>83.38%</td>
<td>90.45%</td>
<td>86.79%</td>
<td>55.80%</td>
<td>88.04%</td>
<td>8.218 G</td>
</tr>
</tbody>
</table>

In the second group, Faster R-CNN MobileNetV3-Large-FPN and Faster R-CNN MobileNetV3-Large-320-FPN are based on Faster R-CNN and MobileNet. The difference is that the input of the latter is resized to 320 × 320 pixels. Although the parameters of the two networks are exactly the same, the computational complexity of the high-resolution model
was seven times that of the low-resolution model. It can be seen from the results in Table 4 that the low-resolution version was completely backward. Considering the results of the SSDlite320 MobileNetV3-Large in the first group, we argue that scaling down the image size leads to poor performance, since the dataset developed in this study can be considered to be a small object dataset, and reducing the resolution means loss of information. The parameters of YOLOv5m are close to those of the previous two networks, but the results were comprehensively better. The PR curves and confusion matrices are shown in Figure 8.

Table 4. The results of the models in Group 2.

<table>
<thead>
<tr>
<th>Model</th>
<th>Training Time</th>
<th>Inference Time</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
<th>mAP (COCO)</th>
<th>mAP (Pascal VOC)</th>
<th>MACs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Faster R-CNN MobileNetV3-Large-FPN</td>
<td>324 s</td>
<td>77.4 ms</td>
<td>57.86%</td>
<td>89.90%</td>
<td>60.35%</td>
<td>34.14%</td>
<td>75.46%</td>
<td>18.410 G</td>
</tr>
<tr>
<td>Faster R-CNN MobileNetV3-Large-320-FPN</td>
<td>214 s</td>
<td>60.1 ms</td>
<td>42.76%</td>
<td>39.81%</td>
<td>41.23%</td>
<td>8.27%</td>
<td>30.97%</td>
<td>2.807 G</td>
</tr>
<tr>
<td>YOLOv5m</td>
<td>90 s</td>
<td>19.0 ms</td>
<td>80.29%</td>
<td>91.30%</td>
<td>85.44%</td>
<td>54.03%</td>
<td>87.87%</td>
<td>24.438 G</td>
</tr>
</tbody>
</table>

Figure 8. PR curves and confusion matrices of models in Group 2: (a) PR curve of Faster R-CNN MobileNetV3-Large-FPN; (b) confusion matrix of Faster R-CNN MobileNetV3-Large-FPN; (c) PR curve of Faster R-CNN MobileNetV3-Large-320-FPN; (d) confusion matrix of Faster R-CNN MobileNetV3-Large-320-FPN; (e) PR curve of YOLOv5m; (f) confusion matrix of YOLOv5m.
The networks in the third group are all single-stage models, and their results are shown in Table 5. The first three models are similar in that their recall is high and their precision is relatively low. As shown in their confusion matrix, all three models yielded more FPs relative to the models in the first two groups. The results of SSD300 VGG16 are more balanced, and the model has the lowest computational complexity. The PR curves and confusion matrices are shown in Figure 9.

Table 5. The results of the models in Group 3.

<table>
<thead>
<tr>
<th>Model</th>
<th>Training Time</th>
<th>Inference Time</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
<th>mAP (COCO)</th>
<th>mAP (Pascal VOC)</th>
<th>MACs</th>
</tr>
</thead>
<tbody>
<tr>
<td>FCOS ResNet50-FPN</td>
<td>403 s</td>
<td>97.8 ms</td>
<td>26.45%</td>
<td>97.07%</td>
<td>41.58%</td>
<td>33.19%</td>
<td>75.50%</td>
<td>125.502 G</td>
</tr>
<tr>
<td>RetinaNet ResNet50-FPN</td>
<td>621 s</td>
<td>95.4 ms</td>
<td>50.09%</td>
<td>93.48%</td>
<td>65.23%</td>
<td>31.71%</td>
<td>72.74%</td>
<td>127.193 G</td>
</tr>
<tr>
<td>RetinaNet ResNet50-FPN v2</td>
<td>665 s</td>
<td>94.7 ms</td>
<td>51.61%</td>
<td>94.44%</td>
<td>66.75%</td>
<td>33.33%</td>
<td>75.90%</td>
<td>128.457 G</td>
</tr>
<tr>
<td>SSD300 VGG16</td>
<td>196 s</td>
<td>76.0 ms</td>
<td>65.66%</td>
<td>76.72%</td>
<td>70.76%</td>
<td>26.36%</td>
<td>64.83%</td>
<td>30.477 G</td>
</tr>
</tbody>
</table>

In Group 4, the MACs values of the Faster-R-CNN-based models were much larger than those of YOLOv5l, and the mAP (Pascal VOC) values were very close, but the mAP (COCO) was slightly lower compared with YOLOv5l. The imbalance between precision and recall was also the main drawback of the first two models, as mentioned for Group 3. The results are shown in Table 6, and the PR curves and confusion matrices are shown in Figure 10.

Figure 9. Cont.
In Group 4, the MACs values of the Faster-R-CNN-based models were much larger than those of YOLOv5l, and the mAP (Pascal VOC) values were very close, but the mAP (COCO) was slightly lower compared with YOLOv5l. The imbalance between precision and recall was also the main drawback of the first two models, as mentioned for Group 3. The results are shown in Table 6, and the PR curves and confusion matrices are shown in Figure 10.

### Table 6. The results of the models in Group 4.

<table>
<thead>
<tr>
<th>Model</th>
<th>Training Time</th>
<th>Inference Time</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
<th>mAP (COCO)</th>
<th>mAP (Pascal VOC)</th>
<th>MACs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Faster R-CNN ResNet50-FPN</td>
<td>657 s</td>
<td>97.3 ms</td>
<td>57.54%</td>
<td>95.52%</td>
<td>71.82%</td>
<td>42.29%</td>
<td>84.61%</td>
<td>133.929 G</td>
</tr>
<tr>
<td>Faster R-CNN ResNet50-FPN v2</td>
<td>848 s</td>
<td>95.3 ms</td>
<td>59.57%</td>
<td>94.88%</td>
<td>73.19%</td>
<td>42.95%</td>
<td>84.91%</td>
<td>280.797 G</td>
</tr>
<tr>
<td>YOLOv5l</td>
<td>146 s</td>
<td>25.3 ms</td>
<td>80.87%</td>
<td>91.23%</td>
<td>85.74%</td>
<td>53.57%</td>
<td>87.65%</td>
<td>32.700 G</td>
</tr>
</tbody>
</table>

Overall, all the models were more accurate in detecting sealed cracks than cracks. We believe that sealed cracks are more clearly characterized than cracks. It also can be seen that the low-resolution models performed worse compared with the high-resolution models. It should be noted that all the PR curves have been truncated because detected instances with confidence scores less than 0.25 and IoUs less than 0.5 have been filtered. All the trained models are publicly available.

Images that never appeared in the training and validation sets were used for testing, in order to understand the capabilities of the different models more intuitively. Some representative results are shown in Figure 11.
Overall, all the models were more accurate in detecting sealed cracks than cracks. We believe that sealed cracks are more clearly characterized than cracks. It also can be seen that the low-resolution models performed worse compared with the high-resolution models. It should be noted that all the PR curves have been truncated because detected instances with confidence scores less than 0.25 and IoUs less than 0.5 have been filtered. All the trained models are publicly available.

Figure 10. PR curves and confusion matrices of models in Group 4: (a) PR curve of Faster R-CNN ResNet50-FPN; (b) confusion matrix of Faster R-CNN ResNet50-FPN; (c) PR curve of Faster R-CNN ResNet50-FPN v2; (d) confusion matrix of Faster R-CNN ResNet50-FPN v2; (e) PR curve of YOLOv5l; (f) confusion matrix of YOLOv5l.
Images that never appeared in the training and validation sets were used for testing, in order to understand the capabilities of the different models more intuitively. Some representative results are shown in Figure 11.

Figure 11. Cont.
Figure 11. Cont.
Figure 11. Cont.
In this study, we proposed a novel dense and redundant annotation method for detecting the structural features of asphalt pavement cracks and sealed cracks. Based on this new annotation method, we developed a dataset containing 10,400 pavement images and made this dataset publicly available for future studies. A semi-automatic method of annotating cracks and sealed cracks was used in order to improve the efficiency of dataset creation. Compared with manual labeling, the semi-automatic labeling method saved 80% of image annotation time, which greatly improved the efficiency of the entire crack detection pipeline. Finally, we tested 13 currently popular object detection models, and the results show that the dense and redundant labeling methods are effective. In summary, the YOLOv5 series models are better and balanced performers among all models, and YOLOv5s is the best one with an F1-score of 86.79% and an inference time of 14.8ms. To conclude, by combining semi-automatic label generation and redundant and dense object annotation with the YOLOv5 series models, we can achieve efficient pavement crack and sealed crack detection.

**Author Contributions:** Conceptualization, methodology, N.Y., Y.L., and R.M.; software, N.Y.; validation, formal analysis, R.M.; writing—original draft preparation, review, and editing, N.Y. and Y.L. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** The data presented in this study are openly available in [https://github.com/CHDyshli/PavementCrackDetection](https://github.com/CHDyshli/PavementCrackDetection) (accessed on 19 August 2022).

**Acknowledgments:** This work was supported by Scientific Research Project of Department of Transport of Shaanxi Province in 2020: Research on Informatization and Application of Highway Pavement.
Disease Treatment Based on IoT + BIM (No. 20-25X). The authors would like to thank Ronggui Ma for excellent technical support and Lin Xiao for critically helping in my life.

Conflicts of Interest: The authors declare no conflict of interest.

References
2. Tran, T.S.; Tran, V.P.; Lee, H.J.; Flores, J.M.; Le, V.P. A Two-Step Sequential Automated Crack Detection and Severity Classification Process for Asphalt Pavements. *Int. J. Pavement Eng.* 2022, 23, 2019–2033. [CrossRef]


