

Article

Collaborative Road Damage Classification and Recognition Based on Edge Computing

Xiaochao Dang ^{1,2,*}, Xu Shang ¹, Zhanjun Hao ^{1,2}  and Lin Su ¹¹ College of Computer Science and Engineering, Northwest Normal University, Lanzhou 730070, China² Gansu Province Internet of Things Engineering Research Center, Lanzhou 730070, China

* Correspondence: dangxc@nwnu.edu.cn

Abstract: Road damage brings serious threats and inconvenience to traffic safety travel. Road damage detection and recognition can assist in eliminating the potential safety hazards in time and reduce traffic accidents. The majority of the existing road damage detection methods require significant computing resources and are difficult to deploy on resource-constrained edge devices. Therefore, the road surface data collected during the driving process of the vehicle are usually transmitted to the cloud service for analysis. However, during the driving process of the vehicle, due to problems, such as network coverage, connection, and response, it is difficult to meet the needs of real-time detection and identification of road damage. Therefore, this paper proposes a road damage classification and identification method based on edge computing. This method adds edge services. First, deep learning models are deployed on edge and cloud servers; then, a standardized entropy is set by information entropy to find the appropriate threshold as well as the best point of edge and cloud that work together to ensure high accuracy and fast response of road damage identification; finally, the cloud uses the data uploaded by the edge to assist the edge in updating the edge model. In comparison with the two cases of uploading data to the cloud server for analysis and uploading data to the edge server for analysis, the results show that the accuracy of the method is 16.21% higher than the method only executed at the edge end, and the average recognition time is 38.82% lower than the method only executed at the cloud end. While ensuring a certain accuracy, it also improves the efficiency of classification and recognition, and can meet the needs of fast and accurate road damage classification and recognition.

Keywords: road damage; edge computing; cloud computing; cloud edge collaboration; deep learning



Citation: Dang, X.; Shang, X.; Hao, Z.; Su, L. Collaborative Road Damage Classification and Recognition Based on Edge Computing. *Electronics* **2022**, *11*, 3304. <https://doi.org/10.3390/electronics11203304>

Academic Editors: Rajendra V. Boppana and Palden Lama

Received: 28 September 2022

Accepted: 11 October 2022

Published: 14 October 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

As well known, road transport is of great significance to the country's economic and social development [1], thus road traffic safety issues have attracted increased attention in recent years. Compared with the causes of traffic accidents, such as drunk driving, speeding, fatigue driving, and vehicle failure, it is more difficult to prevent and control traffic accidents caused by road damage. Since the road surface condition is not always the same, after the road is built and opened to traffic for a period of time, the road surface will slowly appear, such as cracks, pits, loose, subsidence, road damage, and a series of diseases. The emergence of road damage seriously affects the speed and safety of vehicle driving, increases the loss of vehicles, and shortens the service life of the road [2]. Therefore, continuous detection of road damage and timely formulation and arrangement of reasonable plans to repair road damage are of great significance for improving road service life [3], reducing traffic accidents, and ensuring people's travel safety. At present, the detection and identification methods for road damage include walking, human eye observation method, car video screen reading method, etc. [4]. These methods are mainly completed by human eye observation, thus there are problems, such as large error, low efficiency, high cost, and poor safety [5]. As a result, it is of great practical significance to

use advanced technical means to achieve the automatic detection and identification of road damage, improve the detection efficiency of highway maintenance department, and reduce the cost of highway maintenance [6].

Deep learning has achieved great success in many applications, especially in the field of image classification and recognition, which provides a new idea for the detection and recognition of road damage. Therefore, researchers at home and abroad have started to apply deep learning to the detection and recognition of road damage. In [7], an intelligent road damage detection device based on image recognition is proposed, which can detect four types of road damage. The authors of [8] proposed a road crack detection and recognition algorithm based on deep learning and adaptive image segmentation, which can accurately identify cracks on the road surface. In [9], a method of using computer vision to identify road damage is proposed. This method is mainly based on deep learning, which can identify cracks in cement pavement. The authors of [10] proposed a predictive road maintenance strategy based on deep learning, which can provide decision-making and guidance for the repair and maintenance of highway asphalt pavement for a long time. These methods have achieved good results in road damage detection and recognition, and are more accurate, efficient, and safe than traditional manual methods.

Existing methods and models for road damage classification and recognition using deep learning require massive computing resources, thus there is a problem that cannot be deployed on edge servers or edge devices with limited resources. Second, if deployed in a resource-rich cloud, only the road surface data collected during the driving process of the vehicle can be transmitted to the cloud for detection, and then the detection results can be returned by the cloud server. However, it is easily affected by factors, such as network, resulting in poor real-time performance and even inability to detect. Finally, the trained deep learning model is not easy to update after deployment and cannot adapt to the complex and changeable road environment. Therefore, the highway maintenance department needs a pavement damage detection method with fast response, high accuracy, and adaptability to complex and changeable road environment in order to improve efficiency, reduce costs, as well as enable highway maintenance departments to timely and accurately maintain the road. The research of this paper focuses on the challenges and problems of low efficiency and high delay of road damage image classification and recognition method based on deep learning in complex road environment. To solve the aforementioned problems, we introduce the edge server between the user and the cloud server, in order that some computing tasks can be completed by the edge server. To introduce the edge server, we mainly consider the following two parts: First, how to deploy the deep learning model on the edge server with limited resources, we choose herein to prune and compress the deep learning model. Second, how to divide the computing tasks on the cloud server or on the edge server under the condition of ensuring high accuracy. To this end, we choose to determine where the computing task is executed by setting an appropriate threshold. At the same time, we also considered the use of uploaded data to improve the accuracy of the model for better adaptability to the complex and changeable road environment.

Compared with previous work, the main contributions and innovations of this paper are summarized as follows:

- This method provides a faster road damage classification service since we introduce an edge server between the user and the cloud server. The edge server is closer to the user, and the model deployed on the edge server is relatively simple, thus it can provide fast service response.
- This method achieves high accuracy since we set the appropriate threshold through standardized entropy, in which the edge server can identify inaccurate data uploaded to the cloud server for more accurate identification.
- The proposed method can adapt to the complex road environment since we can use the edge server to identify inaccurate data to retrain the edge server.

The rest of this paper is organized as follows: Section 2 reviews the related work. Section 3 introduces the overall framework design of the method. Section 4 describes the

design and implementation of the specific method. Section 5 provides the experimental evaluation. Section 6 summarizes the proposed method and future research directions.

2. Related Work

In recent years, many scholars started to discuss the application of deep learning to road damage classification. Deep learning is a new direction in the field of machine learning research. It adopts a hierarchical structure similar to neural networks. Since LeNet [11] first proposed the use of convolution to solve image classification problems, deep learning has developed rapidly in the field of image classification and recognition, providing a new solution for automatic detection and recognition of road damage. Existing work on road damage identification can be divided into three categories: Cloud-based, edge-based, and edge-cloud-based.

The traditional centralized processing mode based on cloud computing constitutes the user to first upload the collected data to the cloud server. After receiving the data, the cloud server runs the corresponding model for classification, and then sends the results to the user [12]. At present, researchers have proposed many road damage detection models based on deep learning, such as R-CNN [13], YOLO [14], etc. However, to obtain higher accuracy, road damage classification and recognition methods based on deep learning require strong computing power and sufficient storage space. Therefore, to meet the high-precision computing requirements of deep learning models, the majority of the existing methods are based on cloud computing. The authors of [15] proposed a method of automatic detection and location of road damage based on R-CNN algorithm and active contour model, which has high accuracy and can accurately locate road damage. Moreover, the authors of [16] summarized the existing road damage data acquisition methods and described the extraction algorithm of road cracks in detail. In [17], the authors proposed a YOLO-based urban road damage detection method, which can identify eight types of road damage. The authors of [18] proposed a framework based on deep learning, and developed a hybrid model by integrating YOLO and U-Net models to achieve accurate classification and scoring of road damage. Although these methods achieve high accuracy, data collection shows an exponential growth. Cloud servers are far away from the users, which will increase data traffic on the network [19]. Moreover, they are vulnerable to problems, such as network coverage and connection, which will lead to long network transmission delays. Therefore, these methods are not easy to meet the needs of users for rapid response.

Edge computing is a new computing paradigm that can store and execute computations at the edge of a network close to the data source [20]. Users do not need to upload all the data to the cloud, and quickly process some data through nodes deployed at the edge of the network, thus edge computing can provide fast, efficient, and secure services. However, compared with the cloud server, the storage and computing resources of the edge server are relatively limited. Therefore, the deep learning model that requires a large amount of computing storage resources is deployed on the edge. There are two main research directions: The first direction is achieved by reducing the parameters of the model, such as model pruning [21] and compression [22], knowledge distillation, etc. [23]. The authors of [24] proposed the SqueezeNet model, which reduces the model parameters to 1/50 of the original by reducing the number of convolution kernels. Another direction is to combine specific edge hardware to design new algorithms and models [25]. The authors of [26] proposed a lightweight deep neural network model MobileNet using deep separable convolution, which can achieve target detection in edge environment. The authors of [27] designed a computationally efficient CNN architecture called ShuffleNet, which can be deployed at the edge of very limited computing power. Although these methods can deploy deep learning models on edge servers and can meet the needs of users' rapid response, there are some shortcomings in these methods, since these methods sacrifice accuracy while reducing the demand for model resources. Therefore, the accuracy of classification and recognition cannot be guaranteed.

Edge-cloud computing environment is a relatively advanced computing model, which can offload computing tasks to edge servers and divide computing tasks into edge execution or cloud execution [28]. The authors of [29] proposed an application partitioning and scheduling method based on deep neural networks, specifying which tasks are executed locally and which tasks are executed in the cloud. In [30], it is proposed to divide the deep learning model into two parts. One is deployed at the lower level of the edge server, and the other is deployed at the higher level of the cloud server. The edge server loads data from the lower level, and then transmits the data to the cloud server as the input data of the higher level. In [31], the tasks were split, with the edge server tasked with preprocessing the data to remove redundant data and upload it to the cloud server, and the cloud server tasked with receiving and finalizing the data. However, the models proposed by these methods are trained and then deployed to the edge, thus the models are not easy to update dynamically. Finally, some methods only use the edge to preprocess the data or upload the data to the cloud for final processing. Since it is far away from the user, the real-time performance cannot be guaranteed.

3. Method Architecture

This paper proposes a classification and identification method of road damage based on edge computing, which can be used to quickly identify and predict road damage using onboard equipment during vehicle driving. In the architecture proposed in this paper, the edge computing layer is deployed between the cloud computing layer and the in-vehicle devices. The architecture of the system framework is shown in Figure 1, which consists of three layers: Terminal layer, edge layer, and cloud layer. The composition and function of each component are described as follows:

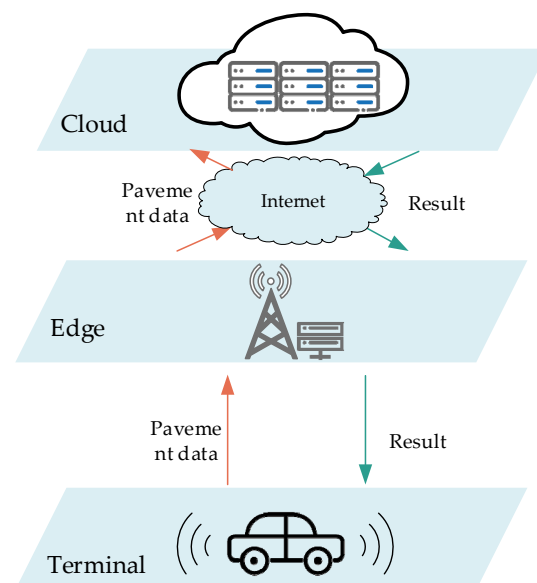


Figure 1. System architecture.

The terminal layer is mainly composed of a series of in-vehicle devices. Vehicle-mounted equipment mainly refers to a series of IoT devices, such as vehicle-mounted radar, driving recorders, and vehicle-mounted cameras, which are responsible for the tasks of capturing data, uploading data, and receiving results.

The edge layer is mainly composed of a group of micro-servers deployed at the data source or at the edge of the network. The deep learning model is deployed on the edge server at the edge layer. Since the edge layer is closer to the user both at the physical level and at the network level, it can ensure low latency and can quickly provide users with real-time result feedback on road disease information.

The cloud layer mainly consists of cloud servers. Since cloud servers have wealthy computing and storage resources, the cloud can be responsible for computing-intensive tasks and assisting in the training of edge layer models.

Since the vehicle travels at a fast speed on the road, this requires a rapid response for road surface detection. Therefore, after the in-vehicle device obtains the road data, it directly uploads the data to the edge server without any processing, and then the edge server runs the deep learning model to detect road diseases. Since the vehicle user is very close to the edge server both at the physical level and the network level, after the edge server receives the data, it runs the deployed deep learning model and then sends the result to the vehicle user. In this way, the classification detection results can be obtained with low delay, and the vehicle user can be warned in time. However, due to the lightweight deep learning model deployed on the edge server, the detection accuracy using the edge server directly may be low. Therefore, this paper uses a threshold to determine whether the edge server can use this feature information for accurate classification and recognition. If it cannot be accurately classified and identified, the edge server will send the data that cannot be accurately classified and identified to the cloud. Let the more complex, higher accuracy deep learning models deployed in the cloud do the processing, with the cloud returning the classification results and using the processed data in turn to train and update the edge servers. The specific process is shown in Figure 2.

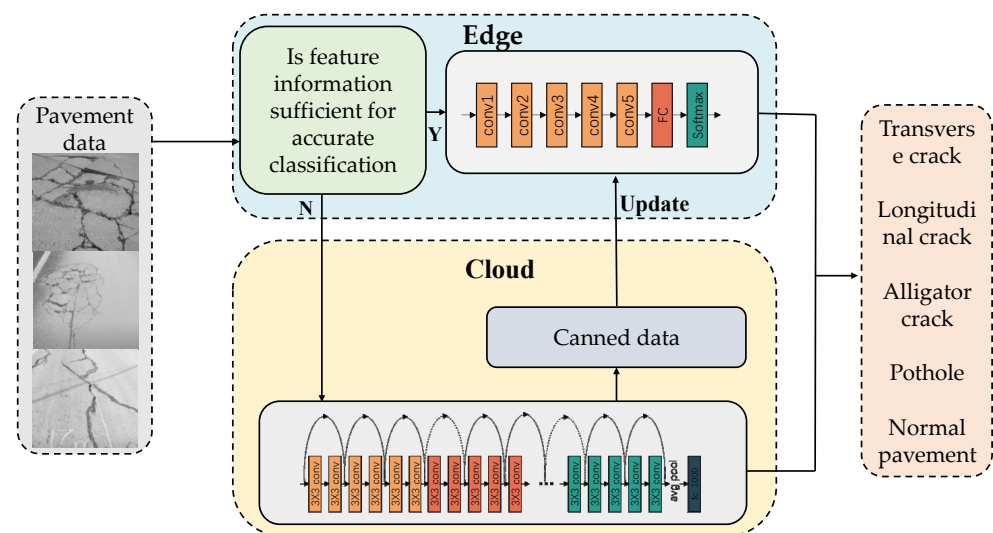


Figure 2. Method flow chart.

In this way, it is possible to classify and identify road diseases accurately and quickly. At the same time, it can use the road data uploaded continuously during the driving process of the vehicle in order that the deep learning model deployed on the edge server can be continuously trained and updated. The accuracy of edge servers will also continue to increase.

4. Method Design and Implementation

4.1. Model Improvement

The AlexNet model was proposed by Hinton et al. [32] in 2012 and won the computer vision recognition challenge. Its model consists of five convolutional layers, three pooling layers, and three fully connected layers and has achieved good results in image classification. The AlexNet model uses ReLU as the activation function to replace the traditional Sigmoid activation function and successfully solves the gradient dispersion problem when the network is very deep. To reduce overfitting, AlexNet model not only uses the data enhancement method, but also uses the dropout method in the first two layers of the full connection layer. Some neurons are omitted randomly, which accelerates the training speed of the model and reduces overfitting. Local response normalization (LRN) is used after

the first, second, and fifth convolution layers to introduce a competition mechanism to local neurons for enhancing the value of a more significant response, while inhibiting other neurons with less feedback to improve the generalization ability of the model. The pooling method uses the overlapping maximum pooling, and the step length of the pooling kernel moving is smaller than the pooling window, which ensures that sufficient features can be extracted.

Although the AlexNet model has achieved good results in image classification by using suitable methods and strategies, there are still some shortcomings. First, the AlexNet model inputs three-channel high-resolution images of $227 \times 227 \times 3$, thus it requires high calculation and storage resources of equipment. Second, the convolution kernel size of the first layer convolution is large, and it is unable to extract more fine image features, resulting in the extraction of incomplete image features; therefore, the convergence rate of the model is slow. Finally, the number of model parameters is huge, up to 61 million model parameters, which greatly increases the difficulty of model training; therefore, it is difficult to deploy the AlexNet model on edge servers with limited resources. Therefore, in order to deploy the model well on the edge server, achieve rapid response and model update. Based on the AlexNet model, this paper conducts lightweight processing, and the improved model network structure is shown in Figure 3.

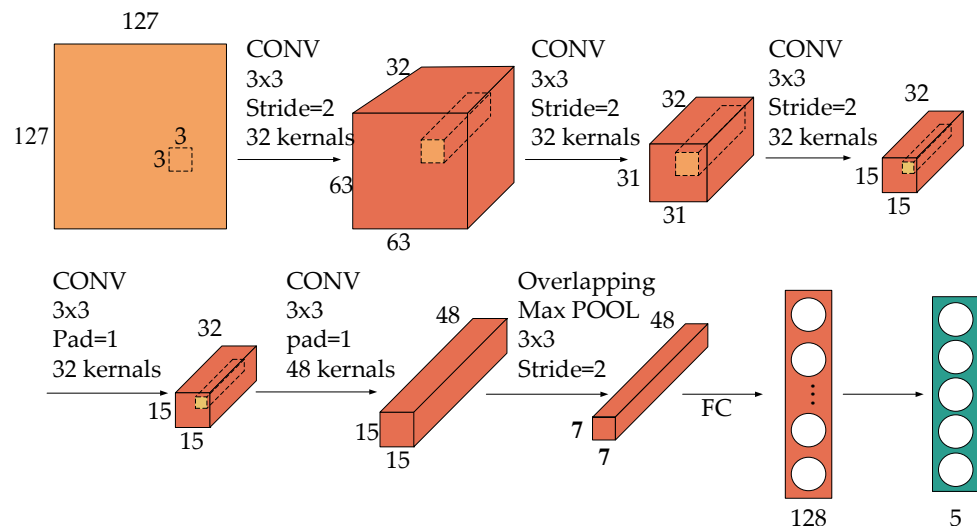


Figure 3. Improved model structure diagram.

The model needs to be lightweight to deploy the model on the edge server with limited resources. Since the different categories of road damage are mainly distinguished by the shape and direction of the cracks, the features required for classification and recognition are fewer, and the image's resolution is not high. Even the low resolution of a single channel can extract effective features, thus based on the AlexNet model, the following methods are used for lightweight achievement. First, the input of the AlexNet model is modified from a $227 \times 227 \times 3$ three-channel high-resolution image to a $127 \times 127 \times 1$ single-channel image, which reduces the size of the input image and reduces the overall number of convolution kernels. At the same time, the requirements for computing and storage resources are reduced. Since the crack features in the road surface image are relatively fine, the convolution kernels of the first two convolutional layers of 11×11 and 5×5 are both modified to 3×3 convolution kernels, in order that the model can extract fine image features. Moreover, as the model only recognizes five types of road damage rather than the original 1000 types of AlexNet, the model removes the first two pooling layers and the first two fully connected layers and only retains the last pooling layer and the first two fully connected layers in the original AlexNet model. A fully connected layer modifies the output dimension of the fully connected layer to 128, making the entire model lightweight. Finally, the dropout method in the fully connected layer is retained

to accelerate model convergence, reduce overfitting, and accelerate model training. The number of road damage categories output by the final model is 5.

Cloud computing and storage resources are rich, thus more layers of neural network models can be deployed in the cloud to obtain better classification results. In fact, with the deepening of the network, the classification accuracy is difficult to improve. ResNet model is a neural network model proposed by He et al. [33] in 2015, which has achieved good results in object recognition and image classification, and the accuracy of ResNet on ImageNet dataset reaches 96.43%. The model solves the model degradation problem with the model network's deepening by adding residual networks in the network. Let x be the input of a network structure, and then the mapping to be solved is $H(x)$. If x is directly used as the final actual output, then the objective function $F(x) = H(x) - x$ represents the error, namely, the residual. The residual network unit is shown in Figure 4.

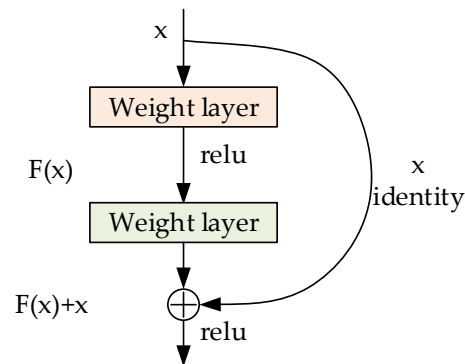


Figure 4. Residual network unit diagram.

With the deepening of the network, when the model degenerates, it will no longer fit $H(x)$. Let the residual $F(x) = 0$, then $H(x) = x$, namely, $H(x)$ is equal to the output x of the previous layer. As the identity mapping is carried out, the network performance will not decline. However, in fact, the residual $F(x)$ is not equal to 0, and new features can be learned by updating the partial weight value of $F(x)$ to improve the performance of the model and avoid network degradation. ResNet model is composed of multiple residual network units, and its activation function is ReLU function. Compared with the traditional VGG network, the complexity of the model is reduced, and the deeper network is used. However, there is no gradient disappearance problem and the classification accuracy is higher. Therefore, this paper only changes the input of ResNet model to $128 \times 128 \times 1$ single-channel image, and then deploys it in the cloud to assist the edge in identifying and retraining road damage.

4.2. Task Partition

To improve the accuracy of the detection results, this paper uses a threshold to decide whether to upload the data to the cloud for detection. For classification tasks, the SoftMax cross-entropy loss function is usually used as the optimization objective. Let y be a true label, x is the input example, and C is the set of all possible labels. For edge-end models, the SoftMax cross-entropy loss function can be written as follows:

$$L(\hat{y}, y; \theta) = -\frac{1}{|C|} \sum_{c \in C} y_c \log \hat{y}_c, \quad (1)$$

where y and z are represented by Equations (2) and (3), respectively:

$$\hat{y} = \text{softmax}(z) = \frac{\exp(z)}{\sum_{c \in C} \exp(z_c)}, \quad (2)$$

$$z = f_{\text{edge}}(x; \theta), \quad (3)$$

The f_{edge} in the formula represents the output of the edge-end model, and θ represents the network parameters of the edge-end model from the input layer to the output layer.

Information entropy is the expected amount of information of the source, and it is a measure of information uncertainty. The higher the entropy value, the greater the uncertainty. Therefore, information entropy can be used to provide a certain judgment basis for task decision-making. According to the information entropy formula, as shown in Equation (4), this paper defines a standardized entropy, expressed by Equation (5), as the confidence criterion.

$$H(y) = -\sum_i y_i \log y_i, \quad (4)$$

$$H(y) = -\sum_{i=1}^{|C|} \frac{y_i \log y_i}{\log |C|}, \quad (5)$$

where C is the set of all possible labels and y is a probability vector containing the vector of computed probabilities for all possible labels. The normalized entropy H has a value between 0 and 1. Close to 0 indicates that the prediction of the sample is more accurate, and close to 1 indicates that the prediction result of the sample is unreliable. The detailed algorithm is shown in Algorithm 1.

Algorithm 1 Threshold value

Input: x, T

Output: $\max \hat{y}$

1. $z = f_{edge}(x)$ // Calculate the output function of the edge
 2. $\hat{y} = \text{softmax}(z)$ // Calculate SoftMax
 3. $H \leftarrow H(\hat{y})$ // Calculate the normalized entropy of the output layer at the edge
 4. *if* $H < T$ *then* // Compare the normalized entropy and threshold
 5. *return* $\arg \max \hat{y}$
 6. *return* $\arg \max \hat{y}$
-

4.3. Edge Server Model Update

The training of data based on deep learning models is very important to improve the performance of mobile applications. This inspires us to retrain models deployed on edge servers through uploaded data to achieve continuous improvement in the accuracy of edge server models. The specific implementation method is as follows. We assume that the results of cloud server classification are 100% accurate. When receiving the data uploaded by the user, if the edge-side determiner judges that the feature information is sufficient for accurate classification and recognition on the edge server, the data are classified and recognized by the edge server, and are no longer uploaded. If the edge-side determiner judges that the feature information is not sufficient for accurate classification and identification on the edge server, the edge server uploads the data to the cloud server. After the cloud server receives the data, it uses a more complex and more refined depth model to predict the data and label them accordingly. Then, the predicted results and labeled data are sent to the edge server. When the data accumulate to a certain amount, the edge server model is retrained to update the edge server model.

In this paper, we use the method of fixing the first few layers, and then using the new dataset to update the weights of the last unfixed layers to achieve the retraining of the model. This can improve not only the accuracy of the edge model, but also save a significant amount of computing resources. When retraining the edge, the first p layers are fixed initially, and then the remaining q layers that are not fixed are retrained. When the newly labelled data accumulate to a certain amount, the edge server uses the labelled data for retraining. Assuming that the newly marked data have accumulated to a certain amount B , the newly marked data are recorded as $\{(\alpha_i, \beta_i)\}_{i=1}^B$. When starting to retrain the deep learning model on the edge server, the original marked data are recorded as A ,

then $\{(\alpha_i, \beta_i)\}_{i=1}^{A+B}$ can be used to retrain, and the edge server is retrained by optimizing the following loss function:

$$K(W_{edge}^q) = \frac{1}{A+B} \sum_{i=1}^{A+B} S_i(f(\alpha_i, W_{edge}^q) \beta_i), \quad (6)$$

where $f(x)$ represents the prediction score of the sample belonging to a certain category, S is a SoftMax function, W_{edge} represents the network parameters of the deep learning model deployed on the edge server, W_{edge}^p represents the fixed network parameters of the first p layer, W_{edge}^q represents the unfixed network parameters of the q layer, W_{edge}^{q*} represents the updated unfixed q layer network parameter, W_{edge}^* represents the value of the updated W_{edge} , and the connector $\cup$ represents the connection of two parameter sets. The specific update phase is shown in Algorithm 2.

Algorithm 2 Model update

Input: $W_{edge}, \{\alpha_i\}_{i=1}^B, \{(\alpha_i, \beta_i)\}_{i=1}^A$

Output: W_{edge}^*

1. The edge server uploads data $\{\alpha_i\}_{i=1}^B$ to the cloud server;
 2. When the cloud server receives data $\{\alpha_i\}_{i=1}^B$, use the deployed deep learning model to obtain label $\{\beta_i\}_{i=1}^B$;
 3. The cloud server sends label $\{\beta_i\}_{i=1}^B$ to the edge server;
 4. $W_{edge}^{q*} \leftarrow \arg \min K(W_{edge}^q)$ Update the network parameters of the unfixed q layer;
 5. $W_{edge}^* \leftarrow W_{edge}^p \cup W_{edge}^{q*}$ Update the network parameters of the edge model;
 6. *return* W_{edge}^*
-

The above architecture is the proposed framework and the detailed process of the framework. In the proposed framework, the cloud and the edge end extend and cooperate with each other, which can effectively reduce the uploading of data to the cloud and reduce the burden and calculation of the network. Moreover, it can provide fast road disease classification and identification service. At the same time, it can continuously improve the detection accuracy and system performance using the data uploaded by users.

5. Experiments

5.1. Experimental Setup

In this section, the performance of the proposed method is verified by experiments. The experimental device is mainly composed of three parts: Vehicle equipment, edge server, and cloud server. The specific configuration is as follows:

Edge server: A Raspberry Pi 4B development board with 8 G of running memory to simulate an edge server;

Cloud server: Emulate a cloud server with a PC with Intel Core i7-10700 CPU and NVIDIA GeForce GTX 1060 Ti GPU.

5.2. Dataset

The experiment in this paper uses road images from different towns and cities in many countries released by the Global Road Damage Detection Challenge 2020 (GRDDC 2020) and some road images collected from Lanzhou City to produce a road damage classification and identification dataset. These images were taken using a smartphone mounted on a car, as shown in Figure 5.



Figure 5. Examples of road images published and collected from Lanzhou by GRDDC 2020.

We screened out five types, including transverse cracks, longitudinal cracks, alligator cracks, potholes, and normal pavement. Remove lanes in the image, water stains, shadows, speed bumps, garbage, debris, street lamps and other factors, and then after flipping, enhanced intercept road damage area. Finally, there are 15,500 images, all of which are grayscale. Further divide the data set into two parts according to the ratio of 7:3, one part is the training set containing 10,850 images, and the other part is the test set containing 4650 images. The models on the edge and cloud are trained with batch 32, momentum 0.9, and learning rate 0.01.

5.3. Experimental Analysis

5.3.1. Threshold T for Partitioning Tasks

Threshold T corresponds to the confidence of whether the data are executed at the edge. Threshold T = 1 indicates that all tasks are executed at the edge, and threshold T = 0 indicates that all tasks are executed at the cloud. Figure 6 shows the relationship between threshold T and task rate at the edge and overall accuracy.

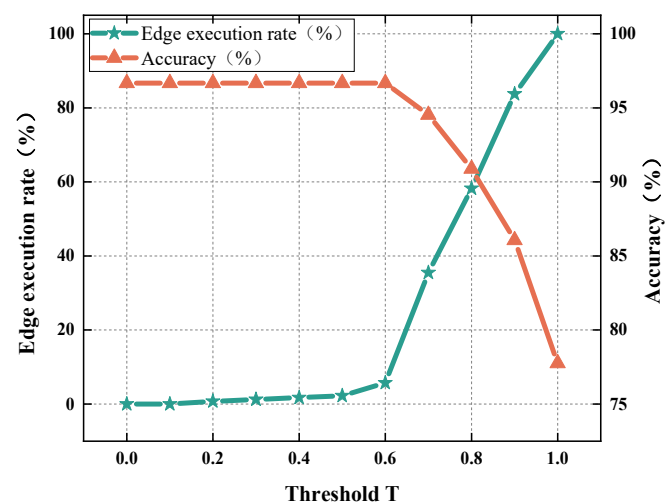


Figure 6. Relationship between threshold T and edge execution rate and accuracy.

It can be observed in Figure 6 that as the threshold T increases, a significant amount of tasks are performed at the edge, and the overall accuracy gradually decreases, which is expected since the accuracy at the edge is usually lower than the cloud. To meet the overall low latency requirement of the system while maintaining the required accuracy, appropriate thresholds need to be set.

The choice of the threshold depends on the application and dataset, and only by choosing a suitable threshold can we find the optimal point where the edge and cloud work together. In Figure 5, it is found that it is best to set the threshold to $T = 0.8$ in this paper, the overall accuracy of the system can reach 91% accuracy, and the edge side performs 58.25% of the processing tasks in order that the majority of the tasks can be performed on edge. Moreover, end-to-end execution ensures fast and accurate classification and recognition. Therefore, a threshold of $T = 0.8$ was set in the subsequent experiments.

5.3.2. Classification Recognition Time

The recognition time is an important indicator to verify the effectiveness of the system. Therefore, to verify the fast response of the method proposed in this paper, this paper measures the inference time on the test set of three different schemes only at the edge, only in the cloud, and using this method, and compares the results. The comparison results are shown in Figure 7.

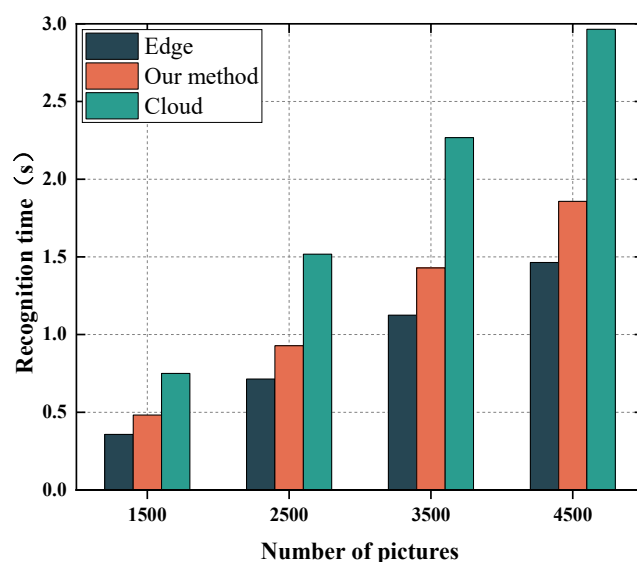


Figure 7. Recognition time for three different scenarios.

From the comparison chart of experimental results, it can be observed that the reasoning time performed only on edge is the shortest, followed by the reasoning time using the method in this paper, and the reasoning time performed only on the cloud is the longest. When the number of test images is 2500, the response time at the edge end can be reduced by 52.94% compared with only performing the task of classification and identification of road diseases in the cloud. The main reason is that the deep learning model deployed in the cloud is more complex and consumes more time due to a large number of complex operations. It can be observed that the more the layers, the longer the inference time, and deploying a simple deep learning model at the edge can quickly respond to user service needs. Compared with only performing the task of classification and recognition of road damages in the cloud, the reasoning time can be reduced by 38.82% using the method proposed in this paper. On the contrary, compared with only performing the task of classification and recognition of road damages on edge, the response time using the method proposed in this paper is only increased by 29.91%, thus it shows that the framework proposed in this paper can provide users with a fast response to the service requirements of road disease identification and classification.

5.3.3. The Performance of Three Schemes, Only at the Edge, the Method in This Paper, and Only in the Cloud

To verify the performance of the three different schemes, namely, only at the edge, only at the cloud, and the method in this paper, the verification index of the binary classification problem is transformed into the multi-classification problem, and the following three indexes are used to evaluate different schemes.

- Accuracy: The ratio of the number of correct classifications to the total amount of data;
- Recall: The ratio of the total number of correct classifications into a certain category to the total number of the category;
- F1 score: The overall evaluation index of precision and recall, as shown in Equation (6):

$$F1 = 2 \frac{\text{accuracy} * \text{recall}}{\text{accuracy} + \text{recall}} \quad (7)$$

The classification and recognition experiments of the three schemes are carried out on the test set. The accuracy of the three schemes for the classification and recognition of various road pavement images is shown in Figure 8.

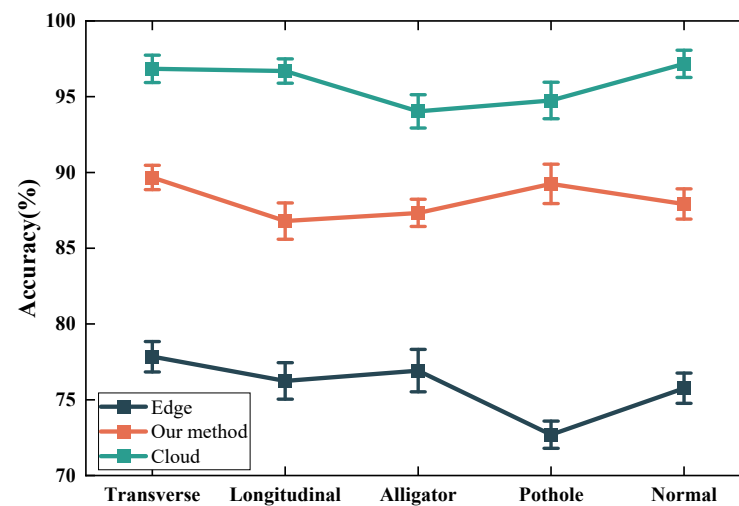


Figure 8. Accuracy on datasets for three schemes: Edge only, our approach, and cloud only.

As can be observed in Figure 8, although the model deployed at the edge is a shallow model, the average accuracy rate of recognition of various road pavement pictures can reach 75%; the model deployed in the cloud has a deeper network structure, So highest average accuracy. In the method proposed in this paper, after the cloud and the edge are coordinated, the average accuracy has been greatly improved compared with only the edge. To further compare and analyze the performance of the three schemes, the recall rate and F1 score of the three schemes for various road pavement pictures were tested in the experiment, and the results are shown in Table 1.

Table 1. Average recall and F1 score for various road surface images on datasets only at the edge, in our method, and in the cloud.

Attributes	Edge	Our Method	Cloud
Transverse crack	77.84%	89.67%	96.84%
Longitudinal crack	75.76%	89.67%	96.70%
Alligator crack	75.70%	89.33%	94.04%
Pothole	69.42%	89.25%	94.75%
Normal pavement	74.33%	89.92%	97.17%
Average recall	74.61%	89.57%	95.90%
F1 score	0.752	0.887	0.959

It can be observed from the above table that the recall rate and F1 value of the implementation scheme only at the edge are the lowest compared with the other two schemes, both below 80%. The method in this paper has a relatively clear performance improvement compared with the method only executed on edge, and various indicators have been greatly improved. Although it reaches about 95% of the method only executed in the cloud, it can also reach 90%. Therefore, the method proposed in this paper has better performance on the test set.

5.3.4. Time to Update the Model

To achieve the dynamic update of the edge model, this paper uses the method of fixing the first few layers and then using the new dataset to update the weights of the remaining layers that are not fixed at the end. Therefore, we will start the training from scratch and compare the training method of this paper on the training set to verify the effectiveness of the model update method on the edge server. The training time of the two methods on the training set is shown in Figure 9.

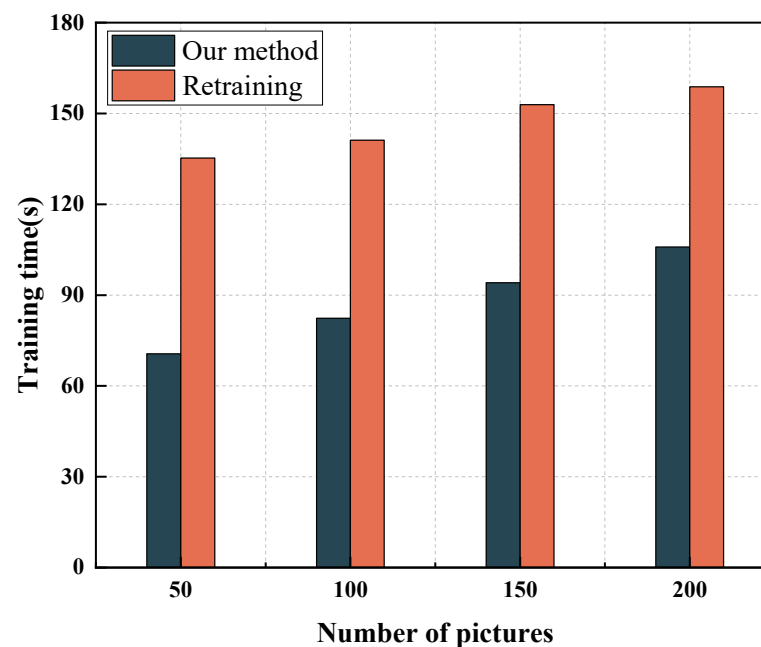


Figure 9. Training time for both methods.

It can be clearly observed that the method of using a fixed first few layers in this paper can reduce the time of updating training by about 40% compared with the method of starting training from scratch. This is due to the fact that the first p layer is fixed during the update training, which indicates that the model inherits the network knowledge of the first p layer during the initial training, in order that the model is trained on the basis of the initial training model during the update and the training time is higher than starting the training from the beginning. Moreover, the method is shorter. Furthermore, when updating the training, since the network parameters of the first p layer are fixed, it is only necessary to fine-tune the network parameters of the model without a fixed rear q layer, in order that the convergence speed of the model will be faster and the training speed will be faster.

5.3.5. Effect of Updating Model on Accuracy

Training data are important to improve the performance of the model. To verify the effectiveness of updating the model with user upload data, the updated edge-end model and the un-updated edge-end model are compared on the dataset. The experimental results are shown in Figure 10.

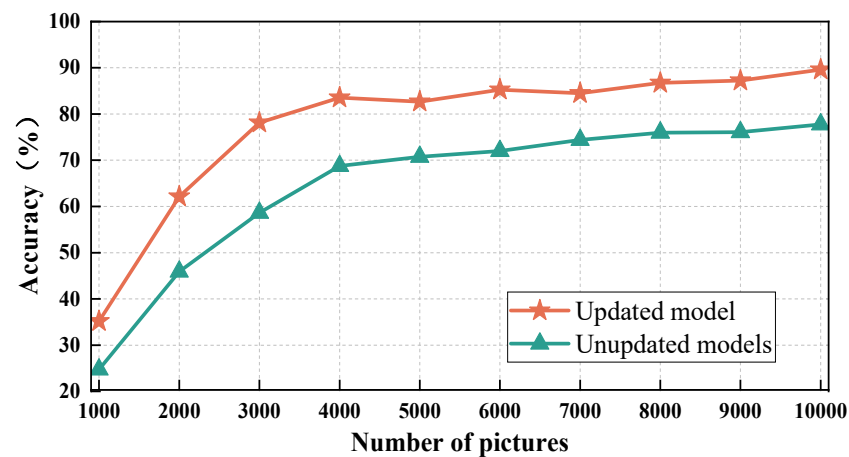


Figure 10. Accuracy of updated and un-updated models.

From the experimental results, with the increase in training data, the accuracy of the two models is improved, which shows that a large number of training data is beneficial to improve the accuracy of deep learning model. At the same time, compared with the un-updated edge-end model, the accuracy of the updated edge-end model is significantly improved. From the experimental results, it can be observed that compared with the un-updated edge-end model, when the training dataset is 1000, the accuracy of the updated edge-end model increases by nearly 10.42%. This is due to the fact that the updated edge-end model has the annotation data provided by the cloud. The un-updated edge-end model has no annotation data, and it can only be the initial accuracy.

5.3.6. The Comparison of this Method with Other Classification Methods

To evaluate the overall performance of our proposed method, in the experimental environment of this paper, we use the same training set for training with the GoogleNet model and the VGGNet-16 model, and compare the accuracy, recall, F1 score, and average recognition time on the test set. The experimental results are shown in Figure 11.

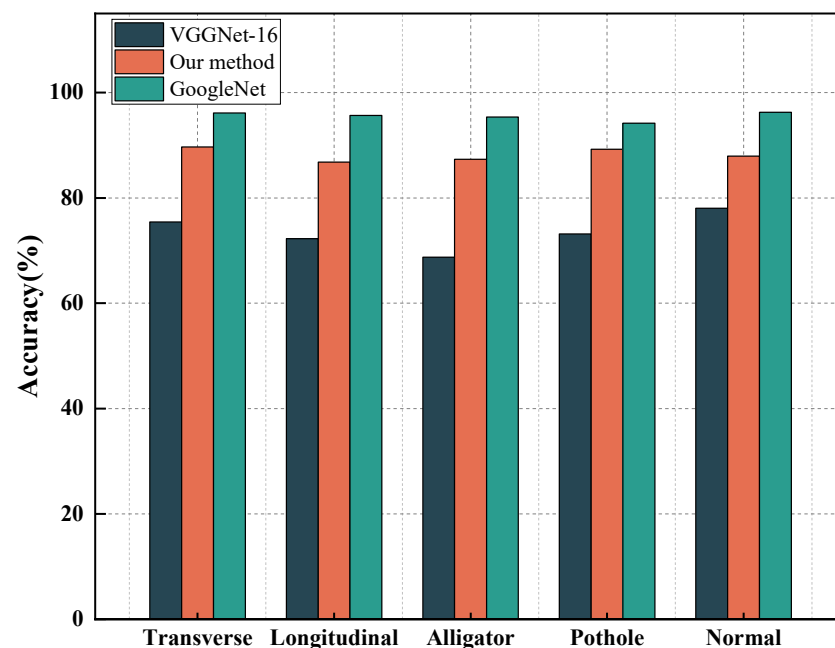


Figure 11. The classification accuracy of this method and the other two classification methods for various types of damage on datasets.

From the classification accuracy of the three classification methods for various types of road damage, it can be observed that the classification accuracy of the GoogleNet model is the highest, and the classification accuracy of various types of road damage images of VGGNet-16 is the lowest. The classification accuracy of the proposed method is between the two and very close to the classification accuracy of the GoogleNet model. It shows that our method can achieve more accurate road damage classification. To further evaluate the performance of the proposed method, the recall rate, F1 score, and average recognition time of the three methods for various road damages are shown in Table 2.

Table 2. The recall rate, F1 score, and average recognition time of this method and the other two classification methods for various types of damage on the dataset.

Attributes	VGGNet-16	Our Method	GoogleNet
Transverse crack	67.7%	89.67%	95.67%
Longitudinal crack	70%	89.67%	94.67%
Alligator crack	50%	89.33%	95.33%
Pothole	70.67%	89.25%	95.59%
Normal pavement	69.42%	89.92%	95.42%
Average recall	65.59%	89.57%	95.34%
F1 score	0.693	0.887	0.954
Average identification time (ms)	4409 ms	482 ms	1273 ms

It can be observed from the above table that the first is the recall rate. From the recall rate of various road damage images, it can be observed that the recall rate of the VGGNet-16 model is poor, and the average recall rate is only 65.56%. The GoogleNet model and the method in this paper also maintain a high recall rate, which can reach about 90% on average. Second, it can be observed from the calculation results of F1 scores that the GoogleNet model and the method in this paper have better results. The F1 score of the method in this paper is 0.887, while the F1 score of the VGGNet-16 model is the lowest, and the result is only 0.693. Finally, it can be observed from the average recognition time that the proposed method is shorter than the other two classification methods. Therefore, it can be observed from the above experimental results that the method proposed in this paper can provide more accurate classification accuracy while quickly classifying and identifying.

5.4. Research Findings and Limitations

This paper designs a method of edge and cloud collaborative computing for the application of road damage image classification and recognition. The main objectives of this research are (1) to solve the problem of high cost and low efficiency of traditional manual road damage identification and classification, (2) to alleviate the problem of increasing network traffic demand, massive data storage, and long response time when a large number of road surface data are uploaded to the cloud server. Through experiments, it is proved that, first, adding edge servers between users and cloud servers can alleviate the pressure on cloud servers. At the same time, it is found that since edge servers are closer to users, they can respond to users' needs faster. Second, it is proved that the method proposed in this paper to find the threshold uses standardized entropy, and then to determine whether the computing task should be divided into edge server or cloud server is feasible. Finally, with the help of the uploaded data and the cloud server, the model on the edge server is adaptive and can be adapted to complex and changeable environments. However, the method of this paper also has limitations. Only the collaboration of a single edge server and a cloud server is considered, which is limited since the number of edge servers is relatively large. In reality, it is impossible to have only a single edge server and a cloud server for collaboration. In addition, the method of this paper ignores the data security and communication costs in data transmission. Therefore, in the next work, more consideration will be given to the coordination and data transmission of multi-edge servers and cloud servers.

6. Conclusions and Future Work

To achieve convenient, fast, and accurate road damage classification and recognition, this paper proposes a cloud-edge collaboration framework for road damage classification and recognition based on edge computing. The edge server is introduced between the user and the cloud server to achieve fast response. According to the deep learning model and dataset, the appropriate threshold can be selected to find the best point where the edge and the cloud work together. In the update phase, the edge-end depth model can be dynamically updated with the help of the cloud, using the uploaded data continuously to achieve higher accuracy and adapt to the changing environment. Moreover, experiments show that when the threshold is set to 0.8 in this paper, the average recognition time of this method is reduced by 38.82% compared with only on the cloud server. Compared with processing only on the edge server, the classification and recognition accuracy of the method in this paper can be improved by 16%. Moreover, compared with the re-start training, the method in this paper can be shortened by about 40%, indicating that the method in this paper can greatly shorten the model update time compared with the re-start training method, and the updated edge server model is compared with the edge server model without update. The average recognition accuracy can be improved by 10.42%. At the same time, compared with several other image classification methods, the method in this paper also achieves better results. The average recall rate can reach 89.57%, the F1 score is 0.887, and the average recognition time is the best. The average recognition time is only 482 ms. These results are discussed in the experimental section and illustrate the findings and limitations of this study. In general, the method proposed in this paper can achieve fast and accurate classification and recognition of road damage, which can meet the needs of highway maintenance departments for fast, accurate, and adaptable classification and recognition of road surface damage. Although this framework is only applied and verified in the classification and recognition of road damage, the framework proposed in this paper has good versatility and is also suitable for other applications based on deep learning models.

There are still some shortcomings in this paper, such as only considering the collaboration of a single edge server and cloud server, ignoring data security, and communication overhead in data transmission. In the future work, we will study the deployment of multiple edge and cloud servers to work together, and incorporate the security and communication costs of data transmission.

Author Contributions: Conceptualization, X.D., X.S. and Z.H.; methodology, X.D. and X.S.; software, X.D., X.S. and L.S.; validation, X.D., X.S. and Z.H.; formal analysis, X.D., X.S. and L.S.; investigation, X.S.; resources, X.D. and Z.H.; data curation, X.D., X.S. and L.S.; writing—original draft preparation, X.S. and L.S.; writing—review and editing, X.D., X.S. and L.S.; visualization, X.D., X.S. and Z.H.; supervision, X.D. and Z.H.; project administration, X.D.; funding acquisition, X.D. and Z.H. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Natural Science Foundation of China, No. 62162056, and the funds for Industrial Support Foundation of Gansu, No. 2021CYZC-06.

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

References

1. Pantuso, A.; Loprencipe, G.; Bonin, G.; Teltayev, B.B. Analysis of pavement condition survey data for effective implementation of a network level pavement management program for Kazakhstan. *Sustainability* **2019**, *11*, 901. [[CrossRef](#)]
2. Mei, Q.; Gül, M.A. Cost effective solution for pavement crack inspection using cameras and deep neural networks. *Constr. Build. Mater.* **2020**, *256*, 119397. [[CrossRef](#)]
3. Oreto, C.; Massotti, L.; Biancardo, S.A.; Veropalumbo, R.; Viscione, N.; Russo, F. BIM-Based Pavement Management Tool for Scheduling Urban Road Maintenance. *Infrastructures* **2021**, *6*, 148. [[CrossRef](#)]
4. Ma, J.; Zhao, X.; He, S.; Song, H.; Zhao, Y.; Song, H.; Cheng, L.; Wang, J.; Yuan, Z.; Huang, F.; et al. Overview of road detection technology. *J. Transp. Eng.* **2017**, *17*, 121–137.
5. Chengjia, H.; Ma, T.; Xu, G.; Chen, S.; Huang, R. Intelligent decision model of road maintenance based on improved weight random forest algorithm. *Int. J. Pavement Eng.* **2022**, *23*, 985–997.

6. Amirhossein, H.S.; Smadi, O. How prediction accuracy can affect the decision-making process in pavement management system. *Infrastructures* **2021**, *6*, 28.
7. Guo, W. Intelligent detection device of pavement disease based on image recognition technology. *J. Phys. Conf. Ser.* **2021**, *1884*, 012032. [[CrossRef](#)]
8. Fan, R.; Bocus, M.J.; Zhu, Y.; Jiao, J.; Wang, L.; Ma, F.; Cheng, S.; Liu, M. Road crack detection using deep convolutional neural network and adaptive thresholding. In Proceedings of the 2019 IEEE Intelligent Vehicles Symposium, Paris, France, 9–12 June 2019.
9. Guo, P.W.; Meng, W.N.; Bao, Y. Automatic identification and quantification of dense microcracks in high-performance fiberreinforced cementitious composites through deep learning-based computer vision. *Cem. Concr. Res.* **2021**, *148*, 12. [[CrossRef](#)]
10. Jiale, L.; Yin, G.; Wang, X.; Yan, W. Automated decision making in highway pavement preventive maintenance based on deep learning. *Autom. Constr.* **2022**, *135*, 104111.
11. Véstias, M.P.; Duarte, R.P.; de Sousa, J.T.; Neto, H.C. Moving deep learning to the edge. *Algorithms* **2020**, *13*, 125. [[CrossRef](#)]
12. Ding, C.; Zhou, A.; Liu, Y.; Chang, R.N.; Hsu, C.; Wang, S. A Cloud-Edge collaboration framework for cognitive service. *IEEE Trans. Cloud Comput.* **2020**, *10*, 1489–1499. [[CrossRef](#)]
13. Xu, X.; Zhao, M.; Shi, P.; Ren, R.; He, X.; Wei, X.; Yang, H. Crack Detection and Comparison Study Based on Faster R-CNN and Mask R-CNN. *Sensors* **2022**, *22*, 1215. [[CrossRef](#)] [[PubMed](#)]
14. Shi, J.; Xu, L.; Chen, J.; Chen, L.; Wang, C. A pavement crack detection system based on Raspberry Pi and YOLOv5 algorithm. *Inf. Technol.* **2022**, *46*, 8.
15. Dongye, C.-L.; Liu, H. A Pavement Disease Detection Method based on the Improved Mask R-CNN. In Proceedings of the 2020 5th International Conference on Information Science, Computer Technology and Transportation (ISCTT), Shenyang, China, 13–15 November 2020; pp. 619–623. [[CrossRef](#)]
16. Wang, W.; Wang, M.; Li, H.; Zhao, H.; Wang, K.; He, C.; Wang, J.; Zheng, S.; Chen, J. Pavement crack image acquisition methods and crack extraction algorithms: A review. *J. Traffic Transp. Eng.* **2019**, *6*, 535–556. [[CrossRef](#)]
17. Alfarrarjeh, A.; Trivedi, D.; Kim, S.H.; Shahabi, C. A Deep learning approach for road damage detection from smartphone images. In Proceedings of the 2018 IEEE International Conference on Big Data (Big Data), Seattle, WA, USA, 10–13 November 2019.
18. Hamed, M.; Adu-Gyamfi, Y.; Buttlar, W.G. Deep machine learning approach to develop a new asphalt pavement condition index. *Constr. Build. Mater.* **2020**, *247*, 118513.
19. Hamid, S.; Bawany, N.Z.; Sodhro, A.H.; Lakhan, A.; Ahmed, S. A Systematic Review and IoMT Based Big Data Framework for COVID-19 Prevention and Detection. *Electronics* **2022**, *11*, 2777. [[CrossRef](#)]
20. Weisong, S.; Hui, S.; Cao, J.; Quan, Z.; Wei, L. Edge computing: A new computing model in the era of Internet of everything. *Comput. Res. Dev.* **2017**, *54*, 18.
21. Tao, L.; Stich, S.U.; Barba, L.; Dmitriev, D.; Jaggi, M. Dynamic Model Pruning with Feedback. *arXiv* **2020**, arXiv:abs/2006.07253.
22. Zhang, Y.; Gao, S.; Huang, H. Exploration and Estimation for Model Compression. In Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada, 10–17 October 2021.
23. Wang, L.; Yoon, K.-J. Knowledge Distillation and Student-Teacher Learning for Visual Intelligence: A Review and New Outlooks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2022**, *44*, 3048–3068. [[CrossRef](#)] [[PubMed](#)]
24. Iandola, F.N.; Han, S.; Moskewicz, M.W.; Ashraf, K.; Dally, W.J.; Keutzer, K. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5 MB model size. *arXiv* **2016**, arXiv:1602.07360.
25. Ning, Z.; Feng, Y.; Collotta, M.; Kong, X.; Wang, X.; Guo, L.; Hu, X.; Hu, B. Deep learning in edge of vehicles: Exploring trirelationship for data transmission. *IEEE Trans. Ind. Inform.* **2019**, *15*, 5737–5746. [[CrossRef](#)]
26. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv* **2017**, arXiv:1704.04861.
27. Zhang, X.; Zhou, X.; Lin, M.; Sun, J. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 6848–6856.
28. Lakhan, A.; Li, J.; Groenli, T.M.; Sodhro, A.H.; Zardari, N.A.; Imran, A.S.; Thinnukool, O.; Khuwuthyakorn, P. Dynamic application partitioning and task-scheduling secure schemes for biosensor healthcare workload in mobile edge cloud. *Electronics* **2021**, *10*, 2797. [[CrossRef](#)]
29. Lakhan, A.; Mastoi, Q.; Elhoseny, M.; Memon, M.S.; Mohammed, M.A. Deep neural network-based application partitioning and scheduling for hospitals and medical enterprises using IoT assisted mobile fog cloud. *Enterp. Inf. Syst.* **2022**, *16*, 1883122. [[CrossRef](#)]
30. Li, H.; Ota, K.; Dong, M. Learning IoT in edge: Deep learning for the Internet of Things with edge computing. *IEEE Netw.* **2018**, *32*, 96–101. [[CrossRef](#)]
31. Liu, C.; Cao, Y.; Luo, Y.; Chen, G.; Vokkarane, V.; Yunsheng, M.; Chen, S.; Hou, P. A new deep learning-based food recognition system for dietary assessment on an edge computing service infrastructure. *IEEE Trans. Serv. Comput.* **2017**, *11*, 249–261. [[CrossRef](#)]

-
32. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *Adv. Neural Inf. Process. Syst.* **2012**, *25*, 84–90. [[CrossRef](#)]
 33. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.