KSCE Journal of Civil Engineering (2024) 28(4):1267-1279
DOI 10.1007/s12205-024-1066-8

pISSN 1226-7988, eISSN 1976-3808
www.springer.com/12205

Hydraulic Engineering

Benchmark Study on a Novel Online Dataset for Standard Evaluation of Deep Learning-based Pavement Cracks Classification Models

Tianjie Zhang[a], **Donglei Wang**[b], **and Yang Lu**[b]

[a]*Dept. of Computer Science, Boise State University, Boise, Idaho 83725, USA*
[b]*Dept. of Civil Engineering, Boise State University, Boise, Idaho 83725, USA*

ARTICLE HISTORY

Received 23 May 2023
Revised 9 September 2023
Accepted 23 November 2023
Published Online 3 February 2024

KEYWORDS

Convolutional neural networks
Pavement cracks
Deep learning
Benchmark study
Crack classification

ABSTRACT

Highway agencies and practitioners expect to have the most efficient method with adequate accuracy when choosing a deep learning-based model for pavement crack classification. However, many works are implemented on their own dataset, making them hard to compare with each other, and also less persuasive and robust. Therefore, a Road Cracks Classification Dataset is proposed to serve as a standard and open-source dataset. Based on this dataset, a benchmark study of fourteen deep learning classification methods is evaluated. Two parameters, the Ratio of F_1 and Training Time (RFT) and Ratio of F_1 and Prediction Time (RFP), are proposed to quantify the efficiency of networks. The results show that ConvNeXt_base reaches the highest accuracy among all models but requires the longest training time. AlexNet takes the least training time among all models, but gains the lowest accuracy. Of the four crack types, the block crack has the lowest accuracy, which means it is the most difficult to detect. SqueezeNet1_0 has the highest efficiency among all models in converting the computing power to accuracy. Wide ResNet 50_2 consumes the longest prediction time among CNN models, while the ConvNeXt_base has the highest feasibility on real-time tasks. To implement a suitable deep learning-based pavement crack inspection, we recommend a good balance between computational cost and accuracy. Based on this, we provide practical recommendations according to different user groups.

1. Introduction

Pavement surfaces are susceptible to various types of cracks that directly impact pavement serviceability and driving safety (Cui et al., 2022; Wen et al., 2022). The inspection and maintenance of roads in the early stage of crack propagation can greatly reduce overall costs (Xu and Liu, 2022). Different crack types are caused by distinct factors, requiring engineers to employ specific techniques for treatment. Thus, pavement crack classification is aligned with the needs of the Department of Transportation (Liu et al., 2020). Classification algorithms play a crucial role in the development of other algorithms such as object detection or segmentation, as they form the foundation. Therefore, the classification of different crack types becomes one of the most important tasks in road maintenance.

Traditional methods like manual counting and crack recognition are labor-intensive and time-consuming (Deng et al., 2019). In order to enhance the efficiency and accuracy of crack classification, researchers have proposed image-based methods such as threshold segmentation (Zhang et al., 2022), edge detection (Han et al., 2021), and minimal path selection algorithms (Kaddah et al., 2019). Although these methods are computationally efficient, they often exhibit poor robustness, particularly in the presence of illumination changes or irregularly shaped cracks. Due to the limitations of image-based methods, deep learning-based models have emerged as the most advanced and popular approach for intelligent pavement crack detection.

Since AlexNet (Krizhevsky, 2014) first showed its potential for image classification tasks in the ImageNet large-scale visual recognition competition in 2012 (Deng et al., 2009), deep learning has become a key advanced detection method in various fields (Zhang et al., 2023b), including pavement crack inspection (Zhang et al., 2023c). The overall process in deep learning does not need manually designed features as traditional machine learning requires.

CORRESPONDENCE Yang Lu yanglufrank@boisestate.edu Dept. of Civil Engineering, Boise State University, Boise, Idaho 83725, USA

It learns from large-scale data which requires little human involvement during training. Numerous works indicated that deep learning, especially Convolutional Neural Network (CNN) has become a popular and powerful tool for road inspection. Gopalakrishnan et al. (2017) developed a modified VGG16 which was pretrained on the ImageNet dataset and the output layer was connected to a linear neural network. This network was trained on 760 images (250 crack images and 510 non-crack images) which were collected from the Long-Term Pavement Performance (LTPP) program, and got an F_1 score of 0.90 in classification. Huyan et al. (2020) proposed a modified U-net network, named CrackU-net, to detect pavement crack images. 3000 pavement images (1695 linear cracks and 1305 map cracks) were collected from highways using high-speed vehicle-mounted action cameras and smartphones, and the model was trained on this dataset. The precision of CrackU-net reached 0.986, which was higher than U-net and Fully-Convolutional Network (FCN). Fei et al. (2019) proposed a CrackNet-V by following the VGG network through reducing the number of parameters and utilizing efficient feature extraction. The proposed network was trained on the Crack Forest Dataset (CFD) which contained 118 urban road surfaces, and got an F_1 score of 0.892. Chen et al. (2021) proposed a deep learning-based thermal image analysis model for pavement detection using a pretrained EfficientNet as the architecture. The model was trained on a self-collected dataset which contained 500 road images and were collected from Liverpool, UK., and it achieved a high damage detection accuracy (0.989). Qu et al. (2020) established a CCD1500 dataset (the images were collected from Google) to train his model which used a combination of LeNet-5 and VGG16 to detect the cracks in concrete pavement. It got an F_1 score of 0.892 on the CFD dataset and 0.901 on the DeepCrack Dataset. Liu et al. (2022a) compared four main variations of typical CNN classification models based on a self-built asphalt pavement crack dataset and found EfficientNet-B3 had the highest accuracy on all types of images tested. His comparison mainly focused on four kinds of structures: MobileNet, ResNet, DenseNet and EfficientNet.

A common drawback of these deep learning-based models is their dataset specificity. They are designed for particular databases, making them prone to failure when applied to different datasets. Hence, evaluating and comparing the performance of these CNN models on a standardized dataset becomes essential. Although there are open-source datasets available for pavement crack segmentation and object detection (such as DeepCrack (Liu et al., 2019) and CRACK500 (Yang et al., 2019)), there is currently a lack of open-source datasets specifically designed for crack classification.

Apart from constructing a standardized dataset, selecting a suitable algorithm for a crack detection task for different users is also an important consideration. Generally, data scientists and computer vision engineers prefer to use deeper convolutional layers to improve the accuracy of deep learning algorithms. The classification accuracy has been significantly improved by novel deep learning structures such as VGG (Simonyan and Zisserman, 2014), GoogLeNet (Szegedy et al., 2015), ResNet (He et al., 2016), DenseNet (Huang et al., 2017), ResNeXt (Xie et al., 2017), ConvNeXt (Liu et al., 2022b). For example, GoogLeNet proposed a new neural network architecture, Inception, to improve the utilization of computing resources and its accuracy. ResNet adopted a residual learning block to ease the training of networks. The ResNeXt network was constructed by repeating the same building block which could reduce the number of hyperparameters. The common characteristics for these models are the deeper layers and large parameters. However, because of the high-demanding requirements of computer hardware, computation time, and the increased volume of data from road maintenance companies, the efficiency of these deep learning algorithms has become a new concern in civil engineering applications. Compared to the computing-intensive deep learning model, efficiency and the mobile model has become a new trend in both computer vision field and the civil engineering field. Wang et al. (Wang and Su, 2021) proposed a lightweight crack segmentation model inspired by Xception and BiSeNet, which showed a good balance between computational cost and performance based on a Crack500 dataset. Hou et al. (2021) proposed a mobile deep learning method, MobileCrack, to classify cracks in asphalt pavement using high classification accuracy inspiration from the classic lightweight CNN model MobileNet (Howard et al., 2019) which was designed for mobile phone CPUs. Zhang et al. (2023a) developed an ECSNet to accelerate real-time pavement crack detection which maintains a good balance between accuracy and efficiency. Que et al. (2023) proposed an integrated generative adversarial network and improved VGG model to automatically classify the asphalt pavement cracks. A lightweight model (or an efficiency model) has fewer parameters and uses less computation power. Thus, it is more suitable for practical application (Wen et al., 2022). The lightweight CNN models, including SqueezeNet (Iandola et al., 2016), MobileNet, MNASNet (Tan et al., 2019), EfficientNet (Tan and Le, 2019) and ShuffleNet (Ma et al., 2018), are designed to work specifically for mobile and resource-constrained situations. For instance, MobileNet proposed a novel layer module which could transfer input from a low-dimensional compressed representation to a high dimension (Sandler et al., 2018). The EfficientNet used a simple yet highly effective compound coefficient to scale all dimensions of depth, width and resolution. ShuffleNet was designed based on the computation complexity. In the civil engineering field, government agencies and practitioners desire to have most efficient classification method with adequate accuracy. Thus, there is a demand to pursue performance-based deep learning pavement crack inspection with a good balance between computational cost and accuracy.

To evaluate and compare the performance of different deep learning-based models for various users, it is crucial to establish a standard dataset. Therefore, this work proposes the Road Cracks Classification Dataset (RCCD) as a publicly available dataset for pavement crack classification tasks. The dataset comprises four balanced types of cracks: Transverse Crack, Longitudinal Crack, Alligator Crack, and Block Crack. Based on this dataset,

a benchmark study of fourteen deep learning-based models is conducted to assess their performance and aid in selecting the most suitable model for specific projects. The evaluation includes accuracy and efficiency metrics, allowing for a comprehensive comparison between computationally intensive CNN models and lightweight alternatives. Recommendations will be provided based on a balanced selection approach, catering to different users, including precision-oriented users, highway agencies, industry practitioners, and lightweight/mobile users.

2. Methods

2.1 RCCD Dataset

In previous studies, researchers often encountered difficulties in comparing different models due to the use of specific datasets. This issue highlights the need for a comprehensive crack classification dataset that can facilitate meaningful comparisons. To address this challenge, we have developed an open-source Road Cracks Classification Database (RCCD), which serves as an empirical basis for crack classification research. The RCCD dataset, available at https://github.com/tjboise/RCCD, was carefully curated by manually cropping images provided by the Federal Highway Administration (FHWA).

The dataset comprises 1600 grayscale images acquired from multiple sources, including Google Street View and ARAN Vehicles, in three different cities: Kansas City, Jefferson City, and Columbia, Missouri. Each image in the RCCD has a resolution of 256 × 256 pixels and contains only one type of crack. The images are annotated into four crack types: Transverse Crack, Longitudinal Crack, Alligator Crack, and Block Crack. Transverse cracks are characterized by cracks that run perpendicular to the direction of the road. Longitudinal cracks, on the other hand, are cracks that run parallel to the direction of the road. Alligator cracks are a type of fatigue cracking and are characterized by interconnected cracks that form a pattern resembling a series of small, irregularly shaped polygons. Block crack is a typically square or rectangular crack that forms a pattern resembling a grid or blocks on the road surface. It often results from the expansion and contraction of the pavement due to temperature changes. Fig. 1 illustrates the distribution of each crack type in the dataset and provides some sample images. The longitudinal and transverse cracks may appear visually similar in some cases, primarily differing in their orientation. However, CNNs can learn specific filters that are sensitive to certain orientations. These filters can help the model differentiate between cracks running in different directions.

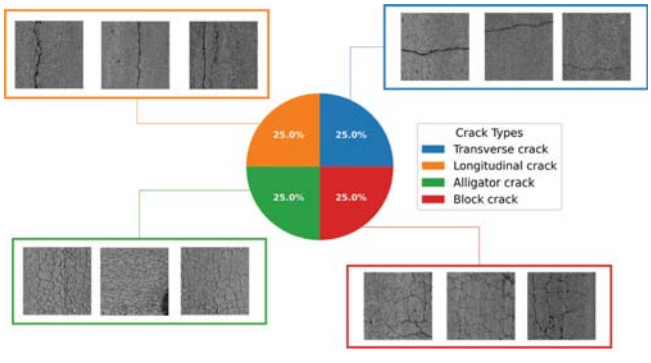

Fig. 1. The Distribution of Each Crack Type in RCCD and Some Crack Samples in Each Kind of Crack

The RCCD has been designed to ensure a balanced representation of each crack type, enabling deep learning models to be trained effectively across different crack types. By utilizing this dataset, researchers can overcome the limitations of small-scale and specialized datasets, facilitating more comprehensive and reliable evaluations of crack classification models.

We provide a Python script in this repository to randomly divide each type crack into training, tuning and testing data sets with a ratio of 6:2:2. The training data is used for training and installation of models. Image augmentation methods including random vertical flip and random horizontal flip are applied during the training procedure when inputting the images to the CNN models. This is because increasing the data amount can improve the accuracy and robustness of the deep learning model. The tuning data is utilized to validate the model and pick up the best-performed model, while the test data is utilized to evaluate the performance of models. As shown in Table 1, a total of 960 training images, 320 tunning images and 320 testing images are used in the training, tuning and testing procedure.

The open-source RCCD would be updated novel photos from various conditions in the future. We welcome users and developers to contribute to this repository. It can provide recommendations for the pavement crack detection community when the algorithms are evaluated on this dataset. We also provided a Python script to let the users easily implement the dataset into their codes.

2.2 Benchmark Study

To demonstrate the applicability of the proposed standard testing hub, we carried out a series of experiments. Moreover, we want to compare some advanced and commonly used deep learning algorithms to figure out their accuracy and efficiency in the pavement crack detection task and provide recommendations to different users, including precision-oriented users, highway agencies, industry practitioners, and lightweight/mobile users.

A benchmark study of fourteen deep learning-based methods is implemented within a single computer code to facilitate the comparison between them. The details of these models are summarized in Table 2. Fourteen classic deep learning neural

Table 1. Training, Tuning and Testing Datasets for CNN Models

Crack type	Training	Tuning	Testing	Total
Transverse crack	240	80	80	400
Longitudinal crack	240	80	80	400
block crack	240	80	80	400
alligator crack	240	80	80	400
Total	960	320	320	1600

Table 2. Summary of the CNN Models Discussed in This Benchmark Study

Models	Year	Params (M)	Model size (MB)	Architecture	
				layers	Basic building blocks
AlexNet (Krizhevsky, 2014)	2014	61.1	233	8	Conv, MaxPool and Dense layers
VGG16 (Simonyan and Zisserman, 2014)	2014	138.4	527	16	Conv, MaxPool, ReLU, Softmax
GoogLeNet (Szegedy et al., 2015)	2015	13.0	49.7	22	Inception
ResNet50 (He et al., 2016)	2016	25.56	97.8	50	Residual Blocks
DenseNet121 (Huang et al., 2017)	2017	7.98	30.8	121	Dense Blocks
SqueezeNet1_0 (Iandola et al., 2016)	2016	1.25	4.77	18	1x1 Conv, Fire Module, Dropout, Global AvePool, Residual Connection, ReLU, MaxPool, Softmax
Resnext50_32 (Xie et al., 2017)	2017	25.03	95.7	50	ResNeXt Block
Wide Resnet 50_2 (Zagoruyko and Komodakis, 2016)	2016	68.88	131	50	1x1 Conv, Conv, Grouped Conv, Global AvePool, ResNeXt Block, Residual Connection, ReLU, MaxPool, Softmax
ShuffleNet v2 (Ma et al., 2018)	2018	2.28	5.28	50	Pointwise Group Conv, Channel Shuffle Operation
MNASNet0_5 (Tan et al., 2019)	2018	4.38	8.59	52	Inverted Residual Block, Conv, deConv
EfficientNet_b0 (Tan and Le, 2019)	2019	5.29	20.4	237	Inverted Bottleneck Residual Block, Squeeze-and-excitation Block
MobileNet v3 (Howard et al., 2019)	2019	5.48	21.1	28	Conv2d, MobileNet v3 Block
RegNet_y_400mf (Radosavovic et al., 2020)	2020	4.34	16.8	50	Conv2d, Squeeze-and-Excitation Blocks
ConvNeXt_base (Liu et al., 2022b)	2022	88.53	338	55	Residual Blocks, Conv2d

networks evaluated in this study, including AlexNet, ResNet50, VGG16, GoogLeNet, DenseNet12, RegNet_y_400mf, EfficientNet_b0, ConvNeXt_base, ShuffleNetv2, SqueezeNet1_0, Resnext50_32, MobileNetv3, MNASNet0_5, and Wide Resnet50_2. We selected these fourteen models to ensure diversity in architecture, optimization techniques, and model size. As shown in Table 2, Resnext50_32 has a unique ResNeXt Block. ResNet50 contains residual modules. This diversity allows us to comprehensively evaluate the performance of deep learning models across a range of design choices and complexities. Also, many of these models have gained significant popularity and have been widely used in various computer vision tasks. This makes them relevant choices for benchmarking and comparison studies. These models are representative of different architectural paradigms, including traditional CNNs (e.g., AlexNet, VGG16), Residual Networks (e.g., ResNet50, Wide ResNet50_2), Inception Networks (e.g., GoogLeNet), and efficient architectures (e.g., EfficientNet_b0, MobileNet v3, RegNet_y_400mf).

As we can see from Table 2, all the models are classic models in the computing vision area rather than the models from the civil engineering area. This is mainly because that most models applied in pavement crack detection are modified from the classic CNN models. Thus, the working theory and performance are similar. Furthermore, most of these modified models in civil engineering are not open source, so it is difficult to repeat the coding work. Therefore, these fourteen classic deep learning neural networks are evaluated in this study. The proposed year of these CNN models is distributed from 2014 to 2022. The whole benchmark evaluating and comparing process is shown in Fig. 2.

During the training process, the accuracy of the model on the tuning data is calculated at each epoch simultaneously. In each epoch, the deep learning structure would update its weights and biases and this updated model would be evaluated on the tunning dataset to get its accuracy. The highest accuracy epoch is chosen among all the epochs. The weights and biases in this epoch are saved as the best-trained model which would be used to evaluate the testing data. By doing this, it can prevent the overfitting problem and improve the robustness of the model. At last, the evaluation results are compared between the different models.

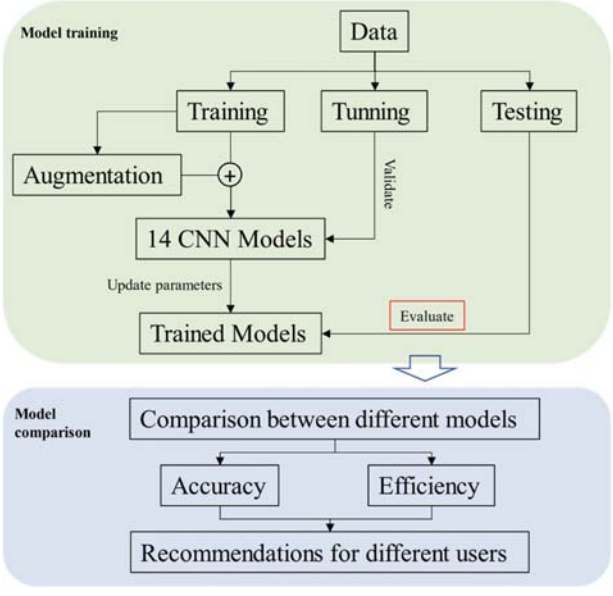

Fig. 2. Schematic View of the Evaluating and Comparing Procedure in the Benchmark Study

To implement the benchmark study in each model, some hyperparameters are modified in the neural networks according to the data. The initial learning rate is set to 0.01 and adjusted during training, where the learning rate decreases by 10% for every 30 steps. Cross entropy is used for the loss function and Stochastic gradient descent (SGD) is utilized as an optimizer to update the network weight with a momentum of 0.4. The batch size of the dataset is set to 32 and each model is trained for 300 epochs. The output classes in the final layer of each model are modified to four classes. These models are all implemented in Python and computed under the following machine speculations: Windows 10, Intel(R) Core (TM) i9-10900X CPU, NVIDIA RTX A4000 with 16 GB memory, and 64GB RAM.

2.3 Evaluation Metrics

In order to evaluate and compare the performance of different deep learning models statistically, each model is trained and tested three times, and each assessment is trained and tested on randomly divided data. The performance of model in each crack type is evaluated by precision, recall, F_1 score, and accuracy, the equations are shown below. The F_1 score is the harmonic mean of precision and recall. It provides a single metric that balances both precision and recall. The harmonic mean gives more weight to lower values, so the F_1 score penalizes models that have a significant imbalance between precision and recall. It ranges from 0 to 1, where one indicates perfect precision and recall, and zero means both precision and recall are at their worst.

$$P = \frac{TP}{TP+FP}, \tag{1}$$

$$R = \frac{TP}{TP+FN}, \tag{2}$$

$$F_1 = \frac{2*(P*R)}{P+R}, \tag{3}$$

$$Acc = \frac{TP+TN}{TP+FP+FN+FP}, \tag{4}$$

where TP is the True Positive, TN is True Negative, FP is False Positive, FN is False Negative, P stands for the Precision, R stands for the Recall, F_1 stands for the F_1 score and Acc represents Accuracy.

The Micro average computes the overall metrics for all classes which will result in a bigger penalization when the model does not perform well with the minority classes compared to the weighted average. Thus, micro average precision, micro average recall, and micro average F_1 score are calculated for each model, the equations are shown below.

$$P_{Macro} = \frac{\sum_1^K P_i}{K}, \tag{5}$$

$$R_{Macro} = \frac{\sum_1^K R_i}{K}, \tag{6}$$

$$F_{1Macro} = \frac{2*P_{Macro}*R_{Macro}}{P_{Macro}+R_{Macro}}, \tag{7}$$

where K is the number of crack types (K = 4 in this work), P_i is the precision value for class i, R_i is the recall value for class i, P_{Macro} stands for the micro average precision, R_{Macro} stands for the micro average recall, F_{1Macro} represents the micro average F_1 score.

For the efficiency evaluation, training time and prediction time are calculated. The training time is counted from the first epoch to the last epoch during the training procedure. Prediction time is calculated when predicting a single image using a trained model. It is used to represent the prediction speed of a model. A faster prediction time means the model can perform better in a real-time crack detection task, as a real-time task requires a quick and in-time prediction. The ratio of macro F_1 score and training time (RFT) is proposed in this work and can be calculated through Eq. (8).

$$RFT = \frac{F_{1Macro}}{training\ time} \tag{8}$$

The RFT shows the efficiency of the model transferring the computing power to good performance in macro F_1 score. A high RFT means that this model can use a small computing source and achieve a high accuracy. It is an important factor for industries whose computing source is limited but want to achieve a relatively high accuracy in road crack inspection.

The ratio of macro F_1 score and prediction time (RFP) is also proposed and calculated to show how much macro F_1 score a model can achieve in a unit predicting time. The RFP value can be calculated using Eq. (9).

$$RFP = \frac{F_{1Macro}}{prediction\ time} \tag{9}$$

This value is an index to show which deep learning-based model can work better in a real-time task. A higher value means that the model uses a shorter prediction time but higher performance in accuracy.

3. Results

3.1 Training Procedure of the Models

Figure 3 shows the relationship between loss and epochs during training process as well as the relationship between accuracy and epochs during the validation procedure of all the models.

As shown in Fig. 3, for CNN models including AlexNet, GoogLeNet, and VGG16, the loss value fluctuates during the training procedure as the epoch increases (Figs. 3(a), 3(f) and 3(m)) It means these models do not learn a lot during the training procedure. In comparison, the loss value decreases and tends to converge as the epoch increases for the models including ConvNeXt_base, ShuffleNet v2, EfficientNet_b0, MNASNet0_5, MobileNet v3, SqueezeNet1_0 and Wide ResNet 50_2. In other words, these models can learn more features from the images and

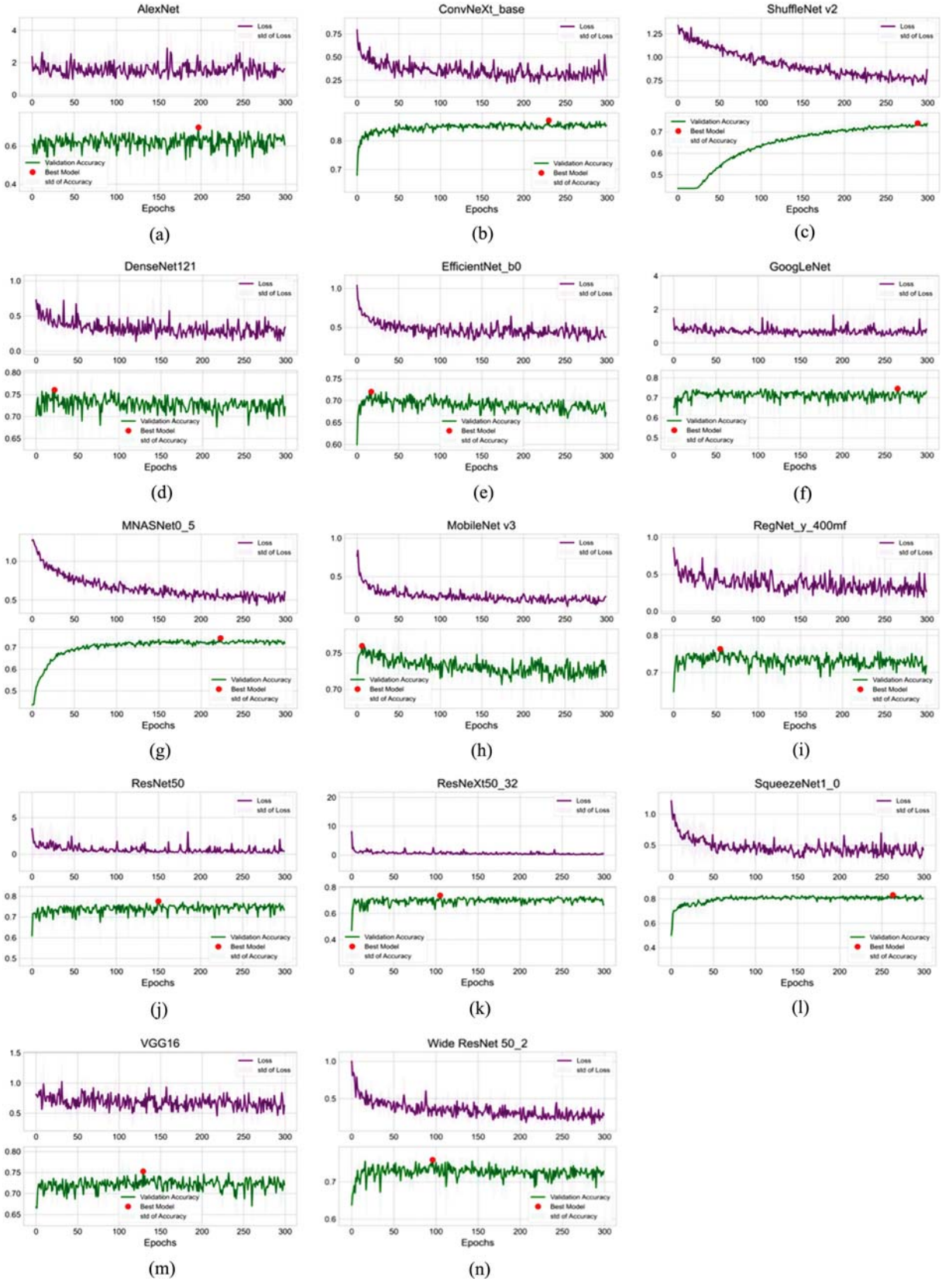

Fig. 3. The Training Loss and Tunning Accuracy of Different Models: (a) AlexNet, (b) ConvNeXt_Base, (c) ShuffleNet v2, (d) DenseNet121, (e) EfficientNet, (f) GoogLeNet, (g) MNASNet0_5, (h) MobileNet v3, (i) RegNet_y_400mf, (j) ResNet50, (k) ResNeXt50_32, (l) SqueezeNet1_0, (m) VGG16, (n) Wide ResNet 50_2

update their parameters more efficiently. The red point stands for the highest performance that happened during the validation process. The parameters in this iteration would be used as the final model to detect the images in the test procedure.

The accuracy curves of tuning data for the fourteen CNN models are also shown in Fig. 3. The red point indicates the best-chosen model which is selected depending on the accuracy of the tuning dataset. This selected model would be tested on the test data to evaluate its performance. Throughout the beginning of the whole training process, the accuracy value of ShuffleNet v2, MNASNet0_5 and ResNeXt50_32 starts out quite low. In contrast, ConvNeXt_base, DenseNet121 and MobileNet v3 begin with relatively high accuracy (about 0.7). However, the accuracy of ShuffleNet v2 and MNASNet0_5 improve rapidly during the training epochs and surpass that of DenseNet121.

3.2 Accuracy Metrics of the Models

The average value (Avg) and standard deviation (Std) of accuracy, macro precision, macro recall and macro F_1 score of all the CNN models are evaluated and summarized in Table 3. As we can see, ConvNeXt_base obtains the highest value in all metrics, followed by SqueezeNet1_0 and ResNet50. It means that ConvNeXt_base has the best performance in accuracy among all CNN models in this crack detection task. It further shows that the macro F1 score, macro recall and macro precision are almost consistent with the accuracy. On the other hand, AlexNet, Resnext50_32 and EfficientNet0_5 obtain the lowest three accuracy values in all models.

By combining and comparing the accuracy metrics and training procedure, the CNN models' performance in the training procedure can be divided into two groups: one is that the loss value fluctuated

Table 3. The Accuracy, Macro Precision, Macro Recall and Macro F_1 Score of CNN Models

Model	Accuracy		Macro Precision		Macro Recall		Macro F_1 Score	
	Avg	Std	Avg	Std	Avg	Std	Avg	Std
AlexNet	0.637	0.050	0.678	0.133	0.616	0.207	0.601	0.094
VGG16	0.720	0.017	0.703	0.044	0.688	0.062	0.691	0.023
GoogLeNet	0.693	0.055	0.694	0.061	0.666	0.113	0.671	0.064
ResNet50	0.747	0.015	0.748	0.085	0.725	0.129	0.723	0.040
DenseNet121	0.730	0.026	0.716	0.061	0.713	0.062	0.708	**0.022**
SqueezeNet1_0	0.813	0.032	0.800	0.075	0.793	0.082	0.790	0.026
Resnext50_32	0.673	0.042	0.683	0.103	0.647	0.184	0.636	0.082
Wide Resnet 50_2	0.740	**0.010**	0.729	0.073	0.693	0.056	0.700	0.028
ShuffleNet v2	0.733	0.021	0.769	0.033	0.658	**0.037**	0.668	0.033
MNASNet0_5	0.713	0.015	0.693	0.036	0.665	0.042	0.676	0.033
EfficientNet_b0	0.687	0.035	0.648	0.051	0.648	0.052	0.645	0.045
MobileNet v3	0.713	0.015	0.698	**0.035**	0.682	0.039	0.688	0.029
RegNet_y_400mf	0.703	0.046	0.686	0.061	0.694	0.086	0.683	0.046
ConvNeXt_base	**0.857**	0.011	**0.858**	0.060	**0.829**	0.073	**0.834**	0.036

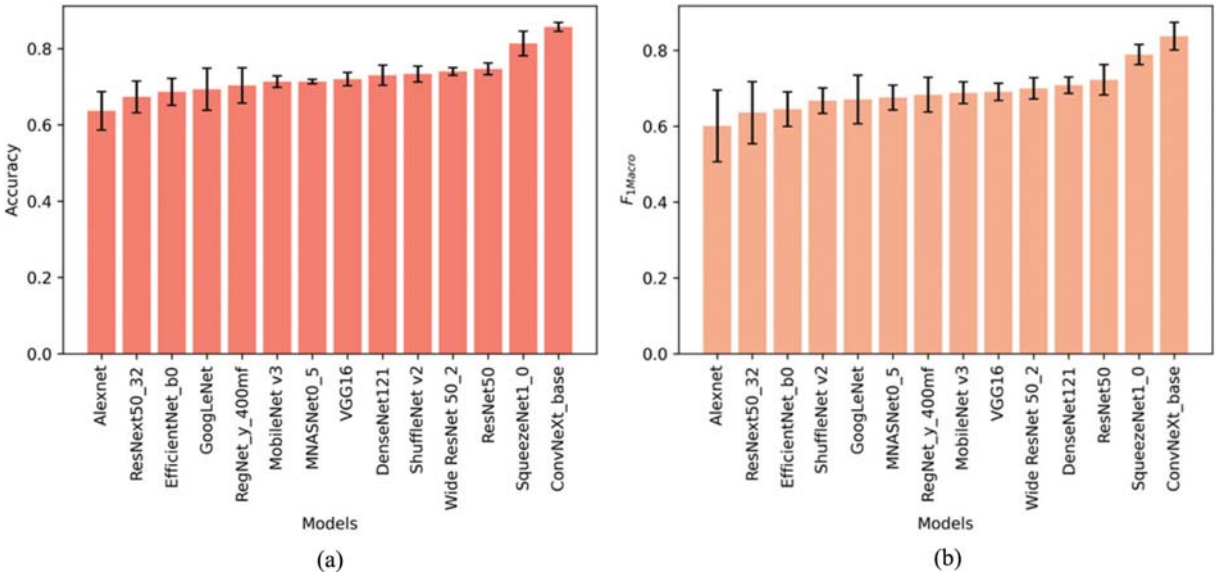

Fig. 4. The Ranked Results of all CNN Models: (a) Accuracy, (b) Macro F_1 Score

during the training procedure as the epoch increase, such as AlexNet, GoogLeNet, VGG16. This kind of model does not perform well in accuracy metrics as the average accuracy of AlexNet, GoogLeNet, and VGG16, is just 0.637, 0.720, and 0.693 respectively. It is mainly due to the fact that during the training procedure, the deep learning structures do not study well about the features of pavement crack images effectively enough. On the other hand, models whose loss decreases with the epochs seem to perform well in accuracy, such as ConvNeXt_base, ShuffleNet v2, and EfficientNet_b0.

The accuracy and macro F_1 score of the CNN models are ranked in the bar plots, respectively which can be seen in Fig. 4.

As we can see from Figs. 4(a) and 4(b), the median value of accuracy and macro F_1 score is both around 0.7. Only ConvNeXt_base and SqueezeNet1_0 get an accuracy higher than 0.8 while for the macro F_1 score, only ConvNeXt_base is higher than 0.8. This suggests that ConvNeXt_base is the most accurate deep learning algorithm for the pavement crack classification tasks when compared to other algorithms. The ShuffleNet v2 ranks 5th in accuracy, however, it ranks 11th in the macro F_1 score. This is mainly because the recall value (0.658) is quite low in ShuffleNet v2, and the macro F_1 score is calculated based on the recall. Recall suggests the level of sensitivity for different crack types. In this crack classification task, a good classifier should perform well in both precision and recall. Thus, the macro F_1 score can be considered a better evaluation than accuracy in certain cases.

3.3 Accuracy Metrics on Different Cracks

After learning the distribution of the overall accuracy metrics of the CNN models, the resulting difference between the four crack types is also studied. Fig. 5 depicts the precision, recall, F_1 score and accuracy distribution of the four kinds of cracks. It apparently shows that the block crack has the lowest metrics among these cracks. This is seen especially in recall value, as the lowest recall of block crack in all models is even lower than 0.2. This means that the model tends to predict cracks as the other three kinds of cracks which can ultimately lead to a lower loss value during the training procedure. In comparison, the evaluation metrics for

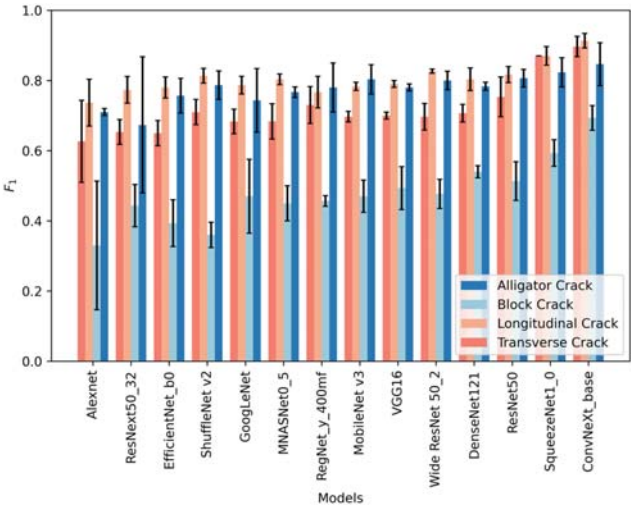

Fig. 6. The F_1 Score Comparison of Four Kinds of Crack in Different CNN Models

Alligator crack, Longitudinal crack and Transverse crack are similar, which means the distribution of models' prediction ability in these cracks is similar. In the boxplot, the green triangle stands for the mean value and the yellow line represents the median value location. As we can see, the median and mean value for Alligator crack and Longitudinal crack are around 0.8, for Transverse crack is around 0.7 while for Block crack, it results in roughly 0.5.

The F_1 score of cracks in different CNN models is calculated and compared as shown in Fig. 6. It is noteworthy that the F_1 score of Block crack is apparently lower than other crack types in each CNN model. The main reason could be that the Block crack is similar to the Alligator Crack. In most CNN models, Alligator crack and Block crack get the two highest F_1 scores. Only in SqueezeNet1_0 and ConvNeXt_base, the F_1 score of the Longitudinal crack and Transverse crack is higher than other cracks. The ConvNeXt_base get the highest F_1 score on all the crack types among the fourteen deep-learning algorithms.

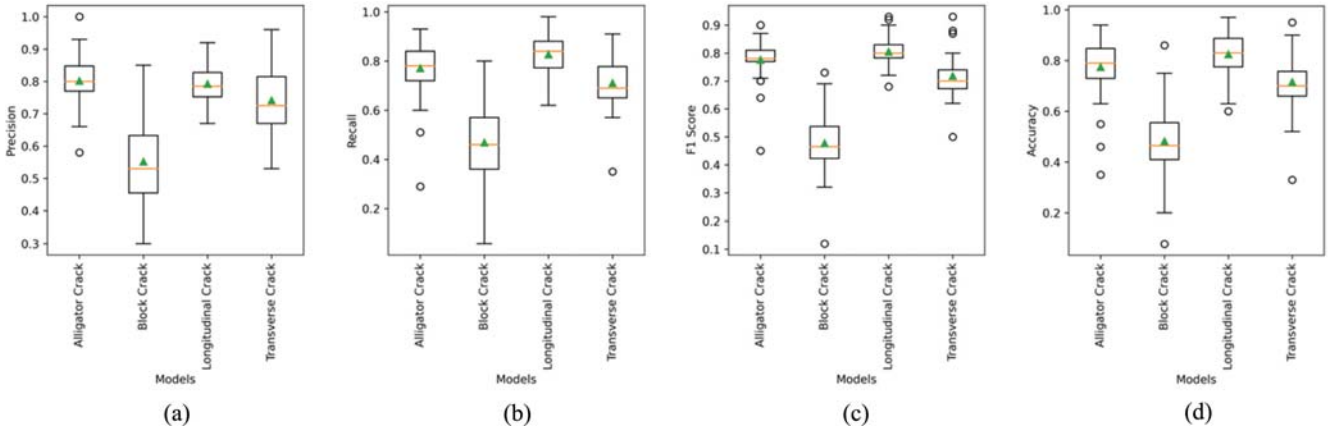

Fig. 5. Box Plots of Four Cracks in Different Metrics: (a) Precision, (b) Recall, (c) F_1 Score, (d) Accuracy

3.4 Efficiency of CNN Models

CNN models aim to use deeper networks to improve the prediction ability. However, crack classification work in civil engineering requires not only the high accuracy of classification but also the speed and efficiency. Crack detection should be real-time work to decrease its influence on transportation flow. Thus, the testing time for each image is an important factor. Because a high-performance GPU is not always equipment in highway agencies, the training time is also considered as a key factor when deploying the CNN models. Advances in neural network efficiency not only improve user experience via higher accuracy and lower latency but also reduce the power consumption to perform identification. In order to evaluate the efficiency of each model, the computing time during the training procedure and the prediction time for each image are recorded in this work which is shown in Figs. 7(a) and 7(b).

As seen in the Fig. 7(a), ConvNeXt_base spends the longest time (3567 seconds) in training, although it gets the highest accuracy and macro F_1 score among all the fourteen models. The AlexNet only spends about 1333 seconds which is the shortest time among all the models, but it also obtains the lowest accuracy on the crack classification task. The training time for CNN models including AlexNet, ShuffleNet v2, MNASNet0_5, SqueezeNet1_0, MobileNet v3, RegNet_y_400mf and GoogLeNet fall within the range of 1300 to 1500 seconds, which are quite low in comparison to other networks. However, a longer training time does not mean a higher accuracy as we can see in Fig. 7(c). The R-square value is only 0.247 for the linear regression between training time and the macro F_1 score, which means that there is no apparent positive linear relationship between training time and test accuracy. The testing time per image is important for real-time detection. As we can see in Fig. 7(b), AlexNet uses the least prediction time among all CNN models. The ConvNext_base uses the second least time for predicting an image although it uses the longest time in training. Eight models use a prediction time less than 0.05s in total.

However, achieving a balance between efficiency and accuracy is crucial for a reliable CNN model. Thus, RFT is used in this study to work as an evaluation metric of trade-off between training efficiency and test accuracy. It is calculated by considering the Macro F_1 score in test data divided by the time consumed during the training procedure, which means how much accuracy a model can get using one unit of computing power. RFP is utilized to work as an evaluation metric of a trade-off between real-time efficiency and test accuracy. It is calculated by considering the Macro F1 score in test data divided by the time consumed in

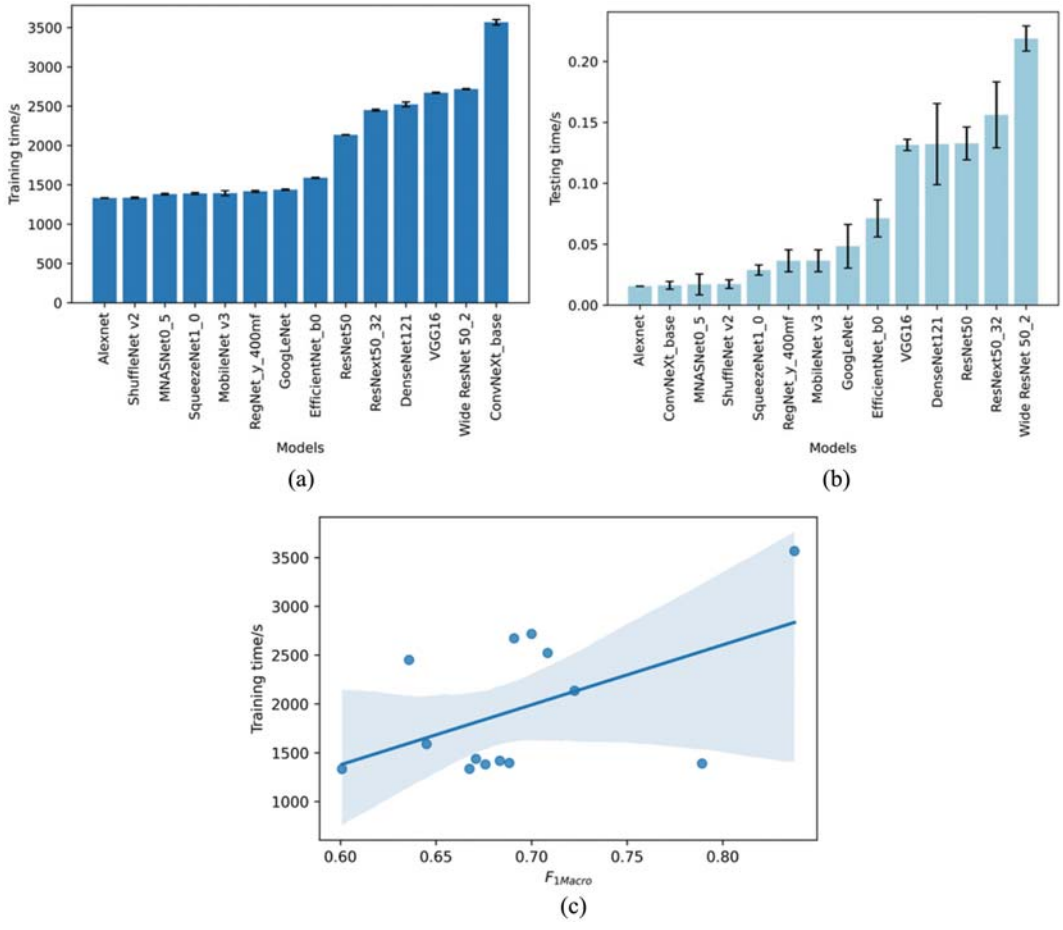

Fig. 7. The Training Time and Prediction Time used in Each Model: (a) The Training Time Used in Each Model, (b) The Prediction Time Needed for a Single Image in Each Model, (c) The Relationship between the Model's Training Time and Test Accuracy

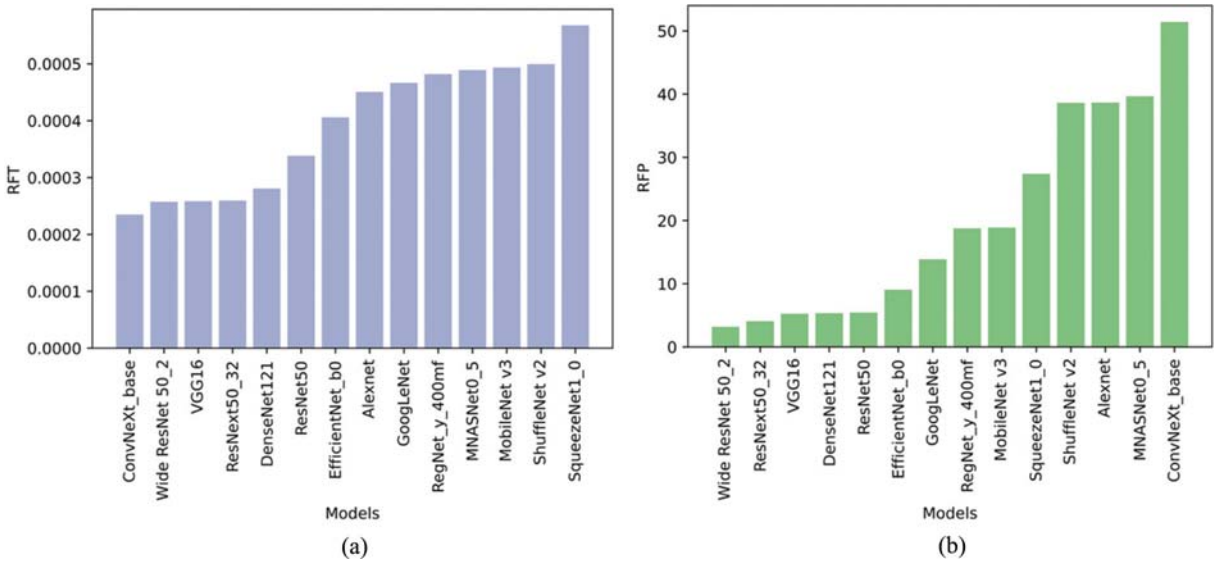

Fig. 8. The RFT and RFP Metrics in Different CNN Models: (a) RFT, (b) RFP

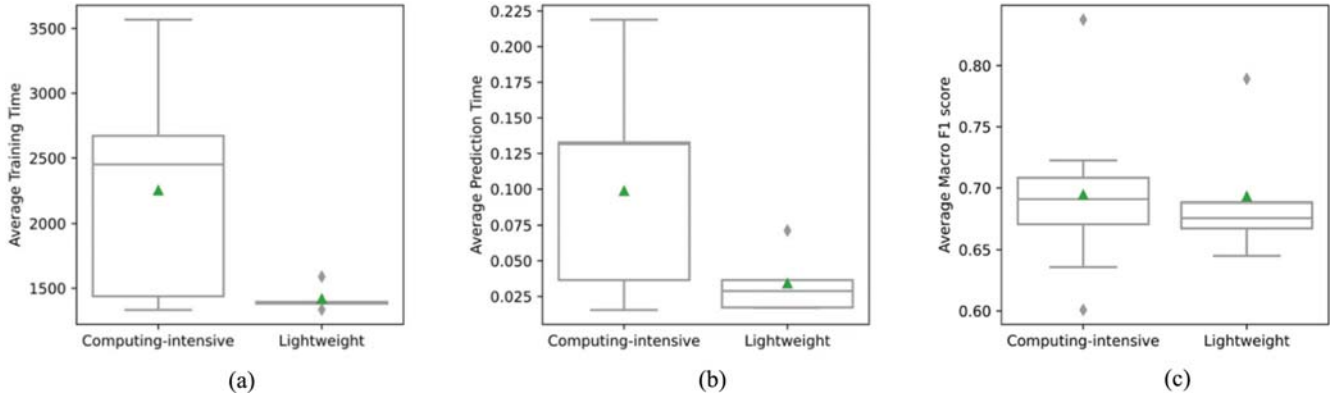

Fig. 9. The Comparison of Lightweight Models and Traditional Models: (a) The Average Training Time during Training Procedure, (b) The Average Testing Time for Each Image, (c) The Average Macro F1 Score in Test Data

predicting an image. Fig. 9 displays the ranked RFT and RFP among different CNN models in this crack classification task.

As we can see from Fig. 8(a), SqueezeNet1_0 has the highest value in RFT while the ConvNeXt_base has the lowest. This is because the SqueezeNet1_0 is one of the only two models which have an accuracy above 0.8, but its training time is very low at only 1390 seconds, similar to the AlexNet time. Although ConvNeXt_base has the highest macro F_1 score among all models, the training time for a ConvNeXt_base model is quite high, which will consume a lot of computing source power.

Figure 8(b) shows the results of RFP in CNN models. If the RFP is high, that means the model can use a small prediction time to get a high accuracy which is very significant in the real-time work. In this case, the ConvNeXt_base get the highest value. AlexNet's macro F_1 score, training time and prediction time are all the lowest among CNN models. However, it ranks third in the RFP.

3.5 Lightweight Model versus Computing-Intensive Model

Traditional CNN models are more concerned about using deeper layers or constructing complex structures to increase the accuracy of models. However, it ignores the requirements for the computing source. The emerging lightweight model has attracted interest due to its small computing time and the low requirement for computing power. Models including SqueezeNet1_0, MobileNetv3, MNASNet0_5, EfficientNet_b0 and ShuffleNet v2 are all lightweight models designed for efficiency. In this benchmark study, the CNN models are divided into computing-intensive and lightweight models depending on the model's parameters. The average training time, prediction time and macro F_1 score are evaluated and compared between these two kinds of models.

As we can see from the Figs. 9(a) and 9(b), the lightweight model has a big difference when compared to the computing-intensive model in terms of computing time. The median value of average training time for lightweight models is lower than 1500 seconds while it is around 2500 seconds for the computing-intensive model. The median value of average testing time for a single road image is about 0.125 for the computing-intensive models while around 0.025 for the lightweight models. That means the lightweight models consume much less computing

time and prediction time than the traditional designed models.

As we can see from the Fig. 9(c), the median and mean value of average macro F_1 score are very close between these two kinds of models. A student t-test (alpha = 0.05) is conducted to compare the mean value of average macro F_1 score in lightweight model and computing-intensive models. The p-value results in 0.74 which means that we cannot reject the hypothesis that the mean value of average macro F_1 score in these two kinds of models is equal. In other words, the lightweight model results in similar accuracy in classifying cracks as the computing-intensive model. According to balanced selection criteria, the lightweight model could be a good choice for mobile users as it consumes less training time and prediction time without decreasing much accuracy.

4. Discussion

An RCCD dataset is first presented in this work as an open-source dataset contributing to standardizing the evaluation of deep learning models in pavement crack classification tasks. Highway agencies and researchers can train and test their models based on this dataset in order to choose a suitable model for their practical application. A benchmark study is conducted to show the performance of state-of-the-art CNN models, which can work as a baseline for other models.

Although there are some important improvements revealed by this work, there are also limitations. The biggest limitation is the representativeness of the RCCD dataset, as it only contains 1,600 images total, which might cause overfitting when training a big model. Also, the images are collected from three cities. More scenarios, including varying lighting, road surfaces, and weather (snow, rainy, and so on) should be included into the dataset to improve its regularization capacity. The lightweight models outperform the computing-intensive models in this dataset, but the lightweight models still contain some drawbacks. For example, lightweight models typically have fewer parameters and a shallower architecture compared to larger, more complex models. This reduced capacity can limit their ability to learn and represent highly intricate or complex patterns in data. Also, they may not generalize as well to diverse or out-of-distribution data and maybe less robust in handling noisy data.

In future studies, more images collected from different scenarios, and various cities would be added into the RCCD dataset to expand its scope, which is beneficial for model training and testing. Moreover, other popular CNN models as well as ensemble learning would be introduced to explore their performance in this dataset. Some code implementations would be done to improve the RCCD's ability to deploy in the existing infrastructure maintenance system. The interpretability of different CNN models will also be learned in the future to gain a deeper insight into the interaction between the model and dataset.

5. Conclusions

In this work, we propose an open-source RCCD dataset as a standard dataset for pavement crack classification tasks. By doing this, we can evaluate and compare deep learning-based models on the same baseline. In order to figure out a proper deep learning method in pavement crack classification for different users, like highway agencies and many industries, a benchmark study is conducted to evaluate the robustness and efficiency of different deep learning architectures based on the proposed dataset. Fourteen classic deep learning-based models are compared and evaluated. The resulting conclusions are summarized below.

1. ConvNeXt_base obtains the highest value in all accuracy metrics (accuracy, macro precision, macro recall and macro F1 score) among all deep learning models, followed by SqueezeNet1_0, and ResNet50. ConvNeXt_base is also the only model that results in all the metrics larger than 0.8.
2. When comparing the accuracy metrics between different cracks, the block crack has the lowest accuracy metrics among cracks in every model. It also indicates that the Transverse crack, Longitudinal crack, and Alligator crack are easier to classify than the Block crack. The ConvNeXt_base gets the highest accuracy and F1 score on all kinds of cracks among all CNN models. This suggests the ConvNeXt_base performs better in identifying all types of cracks in comparison to other models.
3. By comparing the efficiency of the fourteen deep learning algorithms, the ConvNeXt_base consumes the longest training time among models, although it has the highest accuracy and macro F_1 score. Though the highest accuracy model uses the longest training time and the lowest accuracy model, AlexNet, uses the shortest training time, the training time does not indicate a significant linear relationship with test accuracy. Wide ResNet50_2 uses the longest prediction time after the model has been trained, while AlexNet still consumes the shortest predicting time. A model with a short prediction time means it can be applied in a real-time detection task. AlexNet, ConvNeXt_base, MNASNet0_5, ShuffleNet v2, SqueezeNet1_0, RegNet_y_400mf, MobileNet v3, and GoogLeNet are the models whose prediction time is lower than 0.05 s.
4. SqueezeNet1_0 is the most efficient deep learning model on computing power usage, while the ConvNeXt_base is the least training-efficient model. However, in prediction efficiency, the ConvNeXt_base is the most efficient model while Wide ResNet50_2 is the lowest efficient because its prediction time is larger than 0.2.
5. By comparing the lightweight model and traditional model types, it is noteworthy that the lightweight model consumes less training time and prediction time, but the macro F_1 scores made by these two types of models are similar.

Through the benchmark study of deep learning models in pavement crack classification tasks, some recommendations based on balanced selection criteria can be made to different users. For a precision-oriented user, ConvNeXt_base may be chosen as the best model because the accuracy and macro F_1 score are both the highest among all models in this benchmark

study. When computing efficiency comes to the biggest metric, the SqueezeNet1_0 is suggested, as it can decrease the training time dramatically without losing much accuracy. For a mobile user or when the computing power is limited, lightweight models such as SqueezeNet1_0, ShuffleNet v2, MNASNet0_5, EfficientNet _b0, MobileNet v3 can be the first choice as these models use less training time and prediction time than computing-intensive models, but the accuracy does not decrease. For a real-time task, AlexNet, ConvNeXt_base and MNASNet0_5 are the fastest models in real-time predicting, and can be the first choice in a real-time task. If considering both the accuracy and the predicting speed, the ConvNeXt_base could be the best choice as it has the highest value in both macro F_1 score and RFP.

Acknowledgments

We would like to appreciate the Federal Highway Administration (FHWA) for providing the pavement images (https://github.com/UM-Titan/DSPS).

ORCID

Tianjie Zhang http://orcid.org/0000-0002-6697-0777
Donglei Wang http://orcid.org/0000-0002-5888-7439
Yang Lu http://orcid.org/0000-0003-2330-4237

References

Chen C, Chandra S, Han Y, Seo H (2021) Deep learning-based thermal image analysis for pavement defect detection and classification considering complex pavement conditions. *Remote Sensing* 14:106, DOI: 10.3390/rs14010106

Cui B, Wang H, Gu X, Hu D (2022) Study of the inter-diffusion characteristics and cracking resistance of virgin-aged asphalt binders using molecular dynamics simulation. *Construction and Building Materials* 351:128968, DOI: 10.1016/j.conbuildmat.2022.128968

Deng H, Gu X, Wang X, Ww C, Zhu, C (2019) Evaluation of high-temperature deformation of porous asphalt mixtures based on microstructure using X-ray computed tomography. *Construction and Building Materials* 227:116623, DOI: 10.1016/j.conbuildmat.2019.08.004

Deng J, Dong W, Socher R, Li LJ, Li K, Feifei L (2009) Imagenet: A large-scale hierarchical image database. 2009 IEEE conference on computer vision and pattern recognition, Ieee, 248-255

Fei Y, Wang KC, Zhang A, Chen C, Li JQ, Liu Y, Yang G, Li B (2019) Pixel-level cracking detection on 3D asphalt pavement images through deep-learning-based CrackNet-V. *IEEE Transactions on Intelligent Transportation Systems* 21:273-284, DOI: 10.1109/TITS.2019.2891167

Gopalakrishnan K, Khaitan SK, Choudhary A, Agrawal A (2017) Deep convolutional neural networks with transfer learning for computer vision-based data-driven pavement distress detection. *Construction and Building Materials* 157:322-330, DOI: 10.1016/j.conbuildmat.2017.09.110

Han H, Deng H, Dong Q, Gu X, Zhang T, Wang Y (2021) An advanced Otsu method integrated with edge detection and decision tree for crack detection in highway transportation infrastructure. *Advances in Materials Science and Engineering,* 2021, DOI: 10.1155/2021/9205509

He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 770-778

Hou Y, Li Q, Han Q, Peng B, Wang L, Gu X, Wang D (2021) MobileCrack: Object classification in asphalt pavements using an adaptive lightweight deep learning. *Journal of Transportation Engineering, Part B: Pavements* 147:04020092, DOI: 10.1061/JPEODX.0000245

Howard A, Sandler M, Chu G, Chen LC, Chen B, Tan M, Wang W, Zhu Y, Pang R, Vasudevan V (2019) Searching for mobilenetv3. Proceedings of the IEEE/CVF International Conference on Computer Vision, 1314-1324

Huang G, Liu Z, Van Der Maaten L, Weinberger KQ (2017) Densely connected convolutional networks. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 4700-4708

Huyan J, Li W, Tighe S, Xu Z, Zhai J (2020) CrackU-net: A novel deep convolutional neural network for pixelwise pavement crack detection. *Structural Control and Health Monitoring* 27:e2551, DOI: 10.1002/stc.2551

Iandola FN, Han S, Moskewicz MW, Ashraf K, Dally WJ, Keutzer K (2016) SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and< 0.5 MB model size. arXiv preprint arXiv:1602.07360

Kaddah W, Elbouz M, Ouerhani Y, Baltazart V, Desthieux M, Alfalou A (2019) Optimized minimal path selection (OMPS) method for automatic and unsupervised crack segmentation within two-dimensional pavement images. *The Visual Computer* 35:1293-1309, DOI: 10.1007/s00371-018-1515-9

Krizhevsky A (2014) One weird trick for parallelizing convolutional neural networks. arXiv preprint arXiv:1404.5997

Liu F, Liu J, Wang L (2022a) Deep learning and infrared thermography for asphalt pavement crack severity classification. *Automation in Construction* 140:104383, DOI: 10.1016/j.autcon.2022.104383

Liu Z, Mao H, Wu CY, Feichtenhofer C, Darrell T, Xie S (2022b) A convnet for the 2020s. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 11976-11986

Liu J, Yang X, Lau S, Wang X, Luo S, Lee VCS, Ding L (2020) Automated pavement crack detection and segmentation based on two-step convolutional neural network. *Computer-Aided Civil and Infrastructure Engineering* 35:1291-1305, DOI: 10.1111/mice.12622

Liu Y, Yao J, Lu X, Xie R, Li L (2019) DeepCrack: A deep hierarchical feature learning architecture for crack segmentation. *Neurocomputing* 338:139-153, DOI: 10.1016/j.neucom.2019.01.036

Ma N, Zhang X, Zheng HT, Sun J (2018) Shufflenet v2: Practical guidelines for efficient cnn architecture design. Proceedings of the European Conference on Computer Vision (ECCV), 116-131.

Qu Z, Mei J, Liu L, Zhou DY (2020) Crack detection of concrete pavement with cross-entropy loss function and improved VGG16 network model. *Ieee Access* 8:54564-54573, DOI: 10.1109/ACCESS.2020.2981561

Que Y, Dai Y, Ji X, Leung AK, Chen Z, Tang Y, Jiang Z (2023) Automatic classification of asphalt pavement cracks using a novel integrated generative adversarial networks and improved VGG model. *Engineering Structures* 277:115406, DOI: 10.1016/j.engstruct.2022.115406

Radosavovic I, Kosaraju RP, Girshick R, He K, Dollar P (2020) Designing network design spaces. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 10428-10436

Sandler M, Howard A, Zhu M, Zhmoginov A, Chen LC (2018) Mobilenetv2: Inverted residuals and linear bottlenecks. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 4510-4520

Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556

Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A (2015) Going deeper with convolutions. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 1-9

Tan M, Chen B, Pang R, Vasudevan V, Sandler M, Howard A, Le QV (2019) Mnasnet: Platform-aware neural architecture search for mobile. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2820-2828

Tan M, Le Q (2019) Efficientnet: Rethinking model scaling for convolutional neural networks. *International Conference on Machine Learning,* PMLR:6105-6114

Wang W, Su C (2021) Deep learning-based real-time crack segmentation for pavement images. *KSCE Journal of Civil Engineering* 25:4495-4506, DOI: 10.1007/s12205-021-0474-2

Wen T, Lang H, Ding S, Lu JJ, Xing Y (2022) PCDNet: Seed operation–based deep learning model for pavement crack detection on 3d asphalt surface. *Journal of Transportation Engineering, Part B: Pavements* 148:04022023, DOI: 10.1061/JPEODX.0000367

Xie S, Girshick R, Dollar P, Tu Z, He K (2017) Aggregated residual transformations for deep neural networks. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 1492-1500

Xu B, Liu C (2022) Pavement crack detection algorithm based on generative adversarial network and convolutional neural network under small samples. *Measurement* 196:111219, DOI: 10.1016/j.measurement.2022.111219

Yang F, Zhang L, Yu S, Prokhorov D, Mei X, Ling H (2019) Feature pyramid and hierarchical boosting network for pavement crack detection. *IEEE Transactions on Intelligent Transportation Systems* 21:1525-1535, DOI: 10.1109/TITS.2019.2910595

Zagoruyko S, Komodakis N (2016) Wide residual networks. arXiv preprint arXiv:1605.07146

Zhang T, Rahman MA, Peterson A, Lu Y (2022) Novel damage index-based rapid evaluation of civil infrastructure subsurface defects using thermography analytics. *Infrastructures* 7:55, DOI: 10.3390/infrastructures7040055

Zhang T, Wang D, Lu Y (2023a) ECSNet: An accelerated real-time image segmentation CNN architecture for pavement crack detection. *IEEE Transactions on Intelligent Transportation Systems*, DOI: 10.1109/TITS.2023.3300312

Zhang T, Wang D, Lu Y (2023b) Machine learning-enabled regional multi-hazards risk assessment considering social vulnerability. *Scientific Reports* 13:13405, DOI: 10.1038/s41598-023-40159-9

Zhang T, Wang D, Mullins A, Lu Y (2023c) Integrated APC-GAN and AttuNet framework for automated pavement crack pixel-level segmentation: A new solution to small training datasets. *IEEE Transactions on Intelligent Transportation Systems*, DOI: 10.1109/TITS.2023.3236247