Detection of Crack on Asphalt Pavement using Deep Convolutional Neural Network

M. J. A. Ahmad Faudzi¹, M. K. Osman¹, N. A. Muhamed Yusof², K. A. Ahmad³, F. Ahmad³, M. Idris¹, R. A. A. Roaf³, Nor Hazlyna H.⁴

¹ Faculty of Electrical Engineering, Universiti Teknologi MARA Cawangan Pulau Pinang, 13500 Permatang Pauh, Pulau Pinang, Malaysia.
² Electrical Engineering Department, Politeknik Tuanku Sultanah Bahiyah, Kulim, Kedah, Malaysia.
³ School of Computer and Communication Engineering, Universiti Malaysia Perlis, Kampus Pauh Putra, Malaysia.
⁴ School of Computing, Universiti Utara Malaysia, 06010 Sintok Kedah, Malaysia.

E-mail: jadajwad@gmail.com

Abstract. Detection of crack on asphalt pavement is an essential task of monitoring and regulatory inspection. Currently, this task is conducted manually by surveyor or human inspectors for further maintenance works. Manual practice would lead some drawback such as time-consuming, labour intensive, hazardous and also subjective valuation for different individual. To overcome this deficit circumstances an automated technique is implemented. The objective of this study is to develop an intelligent system to detect pavement crack using Deep Convolutional Neural Network (DCNN). This study consists of several procedures which is started with collecting pavement crack images using online and from own developed dataset. The images are pre-processed by resizing the image into desire dimensions. Next, small patches are extracted as inputs to ease of detection and reduce classifier burden. The images further be labelled into two (2) types which is crack and non-crack. In this study, it is utilized Python environment and Keras framework to establish DCNN model. 80% of dataset is used for training set to train, while another 20% is used for testing set to test the model in order to evaluate the performance in terms of accuracy, precision and recall and F1 score. This proposed model is compared on different patch sizes, training algorithms and architectures to get the best classification. Thus, an automated system that able to accurately detect the present of crack in pavement images within speedy computation is successfully developed. To conclude, the system can be used to assist the surveyor or human operator in task of crack detection, so that the process of detection can be done faster and more efficient. This will help in reducing cost of maintenance and enhancing safety of road users.

1. Introduction

Appropriately designed and maintained asphalt pavement can provide several years’ satisfactory service. However, pavement crack is defined as one of common distress problem instead of potholes, ravelling, depression, stripping, corrugation and shoving which can contaminates a good condition of
pavement. Pavement cracks basically can be caused by certain circumstances such as overload from vehicle, overheat by unbalance weather, aging of road structure and others [1]. This kind of distress can lead to more serious of damage if not be settled immediately where can contribute to road accident, property losses and also even worst is deaths. Thus, corrective action including pavement crack detection are compulsory task to maintain a safe driving and is one of the dominant mandates of transportation and regulatory maintenance authorities [2]. Current method is based on manual inspection, means that human inspector needs to travel along the road to detect pavement crack. Since automatic crack detection system are inexpensive, high efficiency and more objective therefore, image processing and machine learning based method have been introduced to assist surveyor in rapid detection of crack [1], [3]-[8]. Not less as well for deep learning which have been applied for numerous researchers to get the better result for this research [2], [9]-[13].

The modernity in this era have been contributed for advance technologies, in order to assist and ease human being for their ordinary assessments. This positive vibe also encouraging numerous researchers from entire world on crafted an automated pavement crack detection system. In earlier research, it is commonly based on advancement of conventional digital image processing such as edge detection, thresholding and mathematical morphology [11]. Thresholding where involves the conversion of each pixel depth level into a binary value which is producing whether white or black [5]. Meanwhile, edge detection work by determining the location of boundaries between crack and its backgrounds in an image. Besides, mathematical morphology is used mathematical algorithm to perform topological and geometrical concept in detecting crack in pavement image. All of these methods are ordinarily based on geometric and photometric assumption to attributes of crack pavement images [11]. To distinguish between crack with non-crack by photometric property, which that crack pixels are darker compare to non-crack pixels. However, by using these approaches it very easily affected to noise since it performed on respective pixels [12]. To handle this issue, method of geometric information is used. For example, crack’s continuity property is deemed to reduce the false detection [3]. Local binary pattern operator is used to determine if a pixel belongs to crack based on local orientation [13]. Wavelet transformation was practiced with multi-scale analysis to separate crack with non-crack regions [11]. In fact, these methods were able to identify cracks effectively, but it is not really precise for detecting entire cracks on the image due to low continuity or high curvature wisely.

For more advancement, the automated crack detection was moved to the next stage of computer vision where utilizing machine learning approach that can be found in many studies. Machine learning is a part of artificial intelligence (AI) that practice of using algorithms to analyse data, learn from data and then perform determination or prediction about new data. This compatible with the objective of machine learning which is to allow the system learns automatically without of human intervention. Most of it are more centralize on extracting feature and recognizing pattern. Linear regression, support vector machine, fuzzy logic and random forest are the most common machine learning techniques practised on pavement crack detection system. Oliveira et al. [6] proposed CrackIT that determines the mean and standard deviation, and unsupervised training method to compare between patches with crack and patches with free crack. The result obtained from the method was compared with manual labelling by human experts. AdaBoost was used by Cord et al. [7] to pick textural descriptors that able to describe crack images. By this method, the textural structure was defined from using a big set of linear and nonlinear filters. It shows a solid textural structure on crack patterns which indicates instability at small scales and a few homogeneities on larger scales. New feature descriptors to identify cracks based on random structured forests was proposed by Shi et al. [8] in CrackForest. The result from conducted experimental showed a better detection in term of precision compared to competing techniques, since the proposed crack descriptor work well to eliminate noises. Nevertheless, the execution of those all mentioned methods are satisfying but it too depends on the extracted features. It not easy to find sufficient features for all pavement with different complicated conditions. Currently, most of researchers focus on using a new technique called deep learning. The method has claimed to produce good accuracy and able to tackle image in various conditions.
Deep learning has been found in many previous literatures to work well compared to conventional machine learning techniques. The main objective of deep learning is also similar to the conventional machine learning which performed regression and classification, but the way and architecture of the techniques are a little bit different. It is relatively reduced pre-processing approach compared to image classification algorithms. For example, the network produces the intact filters that have been tuned from arbitrarily set to extracting features. Whereas, conventional algorithms were manually generated. The advancement of deep learning in recent years implements a novel and promising solution to build an automatic methods of pavement crack detection. Convolutional neural network (CNN) is a type of artificial intelligence method that used the abstraction of deep learning [14]. A CNN usually needs a large number of data during the training process. Cha et al. [10] propose a sliding window to separate the input image into patches and CNN is used to analyse every single patch whether it is content crack or free crack. However, without considering the pixel level, the method was limited to patch level cracks. Zhang et al. [9] implement CNN to determine crack from a single pixel using information obtained from the local patch. Unfortunately, the method disregards the spatial relations between pixels and overrates the crack width. Since CNN can extract good structures from raw data, it be one of the preferred techniques for classification as well as regression application. This paper proposes a CNN based method which can learn the crack feature of a small patch within an input image to analyse the entire crack on pixel level. The main aim of this project is to develop an intelligence system based on DCNN to detect pavement crack in asphalt pavement. In order to achieve that, the first objective is to device a pre-processing method of pavement crack images prior to crack classification. Next, the study models a crack detection system by using DCNN. Lastly, the model is evaluated in detecting pavement crack in term of accuracy, precision, recall and F1 score.

2. Methodology

All the simulation and experiment in this study are conducted using AMD A10-5745M APU processor with Radeon™ Graphics, 2.10GHz CPU and 3.20GB RAM. Only CPU was implemented without applicability of GPU. The network is programed totally in Python version 3.7.3 using Keras framework under TensorFlow banked. In this section, it will be explained the entire procedure and step involved in this study. Basically, this study will be separated into four (4) main steps where start with data collection, pre-processing, development of DCNN architecture and performance evaluation as can be seen in Figure 1. For more details about parameter and specification also be discussed in the following sub-sections.
2.1. Data collection
In general, this study there were using two (2) different datasets for trained and tested which are online and own developed dataset. These two (2) datasets have several main different which are the location of the images were collected where will contributes different condition and surface of the pavement and different image dimension. All information for the datasets is represented in the following sections.

2.1.1. Cracktree200
The online dataset is obtained from the following website: https://github.com/fyangneil/pavement-crack-detection. It consists of five datasets, but only one dataset was selected for this study which is Cracktree200. The Cracktree200 was developed by Zou et al. [3] as dataset for their study. It consists of 206 pavement images with resolution of 800×600 and multiple types of cracks. In this study, a total
of 139 images was selected from the dataset and further resized to 640×480 pixels. The purpose of image resize is to reduce the time duration for pre-processing part and obtain ideal number of patches, which has been selected as 32×32 and 64×64, as shown in Figure 2. Table 1 tabulates this Cracktree200 dataset with two types of patch sizes, renamed as Cracktree32 and Cracktree64. For Cracktree32 it provides total patches of 44,000 and then it be separated into 35,200 for training and 8,800 for testing process, while for Cracktree64 the total patch is 18,600 where 14,880 is use for training and another 3,720 is for testing process. It can simply say that for both data it was separated into 80% and 20% for training and testing process. In additional, normally the amount of crack patches in every image is less than the amount of non-crack patches, so that to equalize the ratio, image augmentation process such as vertical flip, horizontal flip, rotate left and rotate right were applied to increase the number of crack patches. Note that the term positive and negative used in Table 1 and 2 represent the crack and non-crack patches, respectively.

![Figure 2. Positive and negative image for Cracktree200.](image)

| Table 1. The online dataset. |
|-----------------------------|
| **Image size**              | 640 x 480 |
| **Total images**            | 139       |
| **Patch sizes**             | 32 x 32   |
| **Total patches**           | 44,000    |
| **Cracktree32**             |           |
| Training                    | 17,600    |
| Testing                     | 4,400     |
| **Cracktree64**             |           |
| Training                    | 7,440     |
| Testing                     | 1,860     |

2.1.2. **MyDataset**

The second dataset known as MyDataset, is a self-collected dataset. The dataset consists of 98 images pavement images with crack which are obtained using NIKON COOLPIX S6150 digital camera. The image resolution was set to 1024×768 pixels and then extracted into patches with size of 64×64 pixels as presented in Figure 3. These captured images also consist of disturbances such as oil spots, different intensity, shadows, water stains and under different conditions of pavement surface, captured at federal and state roads around Permatang Pauh, Pulau Pinang. The camera is position at 1 meter from the pavement surface. This dataset equally combined of crack and non-crack, consist of 28,000 patches where 22,400 used for training, while other 5,600 is used for testing process. It can simply understand where the data were separated into 80% and 20% for training and testing processes, respectively.

![Figure 3. Positive and negative image for MyDataset.](image)
Table 2. Mydataset.

|                         | Training | Testing |
|-------------------------|----------|---------|
| **Image size**          | 1024 x 768 |         |
| **Total images**        | 98       |         |
| **Patch sizes**         | 64 x 64  |         |
| **Total patches**       | 28,000   |         |
| **Positive**            | 11,200   | 2,800   |
| **Negative**            | 11,200   | 2,800   |
| **Total**               | 22,400   | 5,600   |

2.2. Image Pre-processing
Pre-processing are any process performed on raw data in order to provide ready data to another main process. In this study, some pre-processing techniques have been use including image resizing, image patching, image labelling and also image augmentation to be prepared before training and testing the DCNN model.

2.2.1. Image Resizing
Firstly, image resizing technique is used to resize the original size of raw image. Since this study use of online and own developed dataset, so there are two (2) dimension of raw images which are 800 x 600 and 4068 x 3456 respectively. For online dataset it resized to 1024 x 768 while 640 x 480 for own developed dataset. As in Equation 1, the division value (X) of image dimension divided by patch size must be whole number. Thus, those mentioned dimension size are selected.

\[
X = \frac{\text{Image dimension}}{\text{Patch size}}
\]  

(1)

2.2.2. Image Patching
The second image pre-processing technique that was applied in this study is image patching. Image patching is a process of dividing resized image into multiple patches. In general, the total patches for 640×480 resized image is 75 patches for size of 64×64 and 300 patches for patch size of 32×32 whereas for 1024×768 image resized the total image is 192 patches for patch size of 64×64.

2.2.3. Image Labelling
Next, the data that have been converted into patches be process to the next pre-processing technique called as image labelling. Image labelling technique in this study is used to manually separate between crack and non-crack patches into different folder. Besides, the total crack and non-crack were divided into 80% and 20% which will be save in train and test folder respectively. All patch in those folders were renamed as numeric sequential order begin with zero until the last sequent number using a script written in Python platform.

2.2.4. Image Augmentation
Lastly, image augmentation is used to enlarge the total of patches required to feed the model. There are many techniques can be used in data augmentation for instance zooming, cropping and padding but for this work it only used four which are horizontal flip, vertical flip, right rotate and left rotate. Consequently, by applied these techniques the total original patches will be increase into four time higher than initially.
2.3. Development of DCNN

A typical CNN architecture comprises of input layer, convolutional layer, pooling layer, fully connected layer and output layer. The CNN parameters such as number of layers, size of stride, size of kernel and type of padding are adjusted according to the complexity of data to be solved. This study proposed a DCNN model with three convolution layers and max-pooling layers, and a fully connected layers, as depicted in Figure 4. Meanwhile, Table 3 indicates the list of kernel and stride in every layer as well as their arrangement of DCNN architecture from left to right. The first row represented for layer names followed by number of feature maps while the second row specified kernel size and stride. Extracted patches are utilised as input to the network in both Cracktree200 and MyDataset. All convolutional layers are equipped with kernel size of 3 x 3 and stride of 1, whose credibility has been verified by [10]. In addition, zero padding is implemented on image border in each convolutional layer to maintain the spatial resolution of feature maps after the convolution process. This technique was able to prevent missing of spatial features during the convolution process [15]-[16].

Fully connected layer is placed at the back of model for compiling and combining all the features that have been extracted in prior layer. This proposed network used of two fully connected layers before the output layer of the model. The first layer consists of 256 neurons which are actually the last output of convolution part that have been applied a flatten function while for the second layer consists 64 neurons. The number of neurons at the output layer is equal to two since the number of classes need to be predict in this study is two which are crack and non-crack.

3. Results and discussion

In order to evaluate this proposed DCNN model, it is use of three different datasets which are Cracktree32, Cracktree64 and also MyDataset. The purposed of using three different datasets are to study whether the size of the patch used to train and test the model will affect the model or vice versa. Since MyDataset and Cracktree64 it has a same size of patches which is 64×64 pixels, make them larger in term of memory and dimension compared to Cracktree32 with size of 32×32 pixels. Next, to observe did the different condition and area of developed dataset will affect the progress of model or not.
3.1. Results of crack detection using

3.2. Cracktree200 dataset
Table 4 shows performance comparison between the proposed method with Zou et al. [3]. In [3], the CrackTree method combine with shadow-removal algorithm (GSR) to classify crack and non-crack. The result for precision, recall and F1 score for CrackTree method are taken from [3] whereas for accuracy and loss it not displayed. The comparison can be made since these two methods are utilized the same dataset which is Cracktree200. It can be seen on the results the proposed DCNN has outperformed the CrackTree method with approximately 0.12 for F1 score. On contrary for the recall, CrackTree produced better result but still cannot overcome the score of the proposed DCNN model.

| Method               | Zou et al. [3] | Our proposed method |
|----------------------|----------------|---------------------|
| Accuracy             | -              | 0.9699              |
| Precision            | 0.7900         | 0.9679              |
| Recall               | 0.9200         | 0.9620              |
| F1                   | 0.8500         | 0.9699              |
| Loss                 | -              | 0.1066              |

Table 5 indicates the performance evaluation results of proposed crack detection model based on Cracktree200 dataset that have been divided into two (2) which are Cracktree32 and Cracktree64. The accuracy, precision, recall and F1 score using 32×32 size of patch produced higher results compared to 64×64. Meanwhile, for value of loss between these two (2), Cracktree32 indicates a lower value. In other aspect, it was noticed that the bigger patch size will yield higher time taken to complete the model. Therefore, from this analysis it noticed that the smaller the patch size the easier model to learn and will produce a better result. However, since the result is not large different gap, the next analysis is used Cracktree64 as input of model.

| Dataset      | Cracktree32 | Cracktree64 |
|--------------|-------------|-------------|
| Accuracy     | 0.9751      | 0.9699      |
| Precision    | 0.9793      | 0.9679      |
| Recall       | 0.9712      | 0.9620      |
| F1           | 0.9752      | 0.9699      |
| Loss         | 0.0960      | 0.1066      |
| Time Taken   | 1:17:18     | 1:50:10     |

Table 6 indicates the performance evaluation results of different optimizer or also known as training algorithm. For this section, the model is set for different training algorithms which are Adaptive Moment Optimization (Adam), Root Mean Square Propogation (RMSprop) and Stochastic Gradient Descent (SGD). Each of them was build up by different algorithm to update the weight parameters in order to minimize the loss function. This proposed DCNN model is found to be better with Adam and RMSprop but significantly poor on SGD. In addition, it can be seen from Table 6 that the training algorithms has different duration taken by the model to finish the training and testing process. Adam was found be the fastest training algorithm as compared the other two with the highest accuracy and minimum loss.
Table 6. Crack detection evaluation results using different training algorithms.

| Training algorithm | Adam     | RMSProp  | SGD      |
|---------------------|----------|----------|----------|
| Accuracy            | 0.9699   | 0.9513   | 0.7473   |
| Precision           | 0.9679   | 0.9624   | 0.6231   |
| Recall              | 0.9620   | 0.9416   | 0.8290   |
| F1                  | 0.9699   | 0.9519   | 0.7115   |
| Loss                | 0.1066   | 0.1919   | 0.5263   |
| Training Time       | 1:50:10  | 2:32:20  | 3:00:46  |

Table 7 presents the performance evaluation results of different CNN architecture. Compared to single, two (2) and four (4) layers, 3 layers produces most high accuracy and lowest loss. Thus, it is clearly be seen that the three layers convolution part for the proposed DCNN model is most suited and outperforms others. The result with four (4) layers is also high where only 0.0045 accuracy behind three (3) layers but the time consumed is too big. The study have been conducted and found that the model with over complexity will cause an overfitting but if very low complexity it will cause underfitting [13], [14]. Therefore, the architecture of model must be suitable with difficulty of classification problem, in order to get the best performance of the DCNN model.

Table 7. Crack detection evaluation results using different patch CNN architecture.

| Number of layers | 1       | 2       | 3       | 4       |
|------------------|---------|---------|---------|---------|
| Accuracy         | 0.7462  | 0.8825  | 0.9699  | 0.9654  |
| Precision        | 0.5527  | 0.8468  | 0.9679  | 0.9588  |
| Recall           | 0.9017  | 0.9120  | 0.9620  | 0.9434  |
| F1               | 0.6853  | 0.8782  | 0.9699  | 0.9510  |
| Loss             | 0.9441  | 0.3397  | 0.1066  | 0.2136  |
| Time Taken       | 2:35:34 | 2:52:38 | 1:50:10 | 2:16:08 |

3.3. Result of Crack Detection using Real Data
After determining the most ideal characteristics for the proposed DCNN model finally, it be tested on real data which is MyDataset that have been developed during this study and the structure of the pavement image is general pavement condition used in Malaysia. Based on Table VIII, the evaluation result that was produced is quite excellent with accuracy of 0.9827 and followed by very low loss which is 0.0527 by only 10 epochs. Nevertheless, to complete train and test for MyDataset it took about 3 hours and 32 minutes which is very long computational caused by a very large dataset with total of 28,000 input data and 64×64 size of patch. This also influenced by external cause which are low specification of processor and compute only using CPU instead of GPU. Next, for Figure 5 and 6 illustrate the training and evaluation graph for accuracy and loss respectively. Some researchers are use validation and testing separately, however in this study, the validation and testing are the same. It can be observed that the training process required less epoch to reach the highest accuracy and minimum loss. After the 4th epoch, it will be no significant change in accuracy and loss function. This is show that a proposed DCNN model is faster to learn by only less number of epochs. Furthermore, Figure 7 shows the confusion matrix result of proposed DCNN based on MyDataset. It achieves a good classification scores with a total of 2671 patches are correctly predicted as positive patches (TP) and the rest of 42 patches were wrongly predicted (FP). Then as much as 2770 patches are correctly predicted as negative patches (TN) and the rest of 30 patches were wrongly predicted (FN) over the total 2,800 positive and negative of the test set.
Table 8. Crack detection evaluation results using MyDataset.

| Metric     | Value   |
|------------|---------|
| Accuracy   | 0.9827  |
| Precision  | 0.9775  |
| Recall     | 0.9877  |
| F1         | 0.9826  |
| Loss       | 0.0527  |
| Time Taken | 3:32:28 |

Figure 5. Training and testing accuracy against number of epochs.

Figure 6. Training and validation loss against number of epochs.
4. Conclusion
A DCNN with three convolution layers was proposed and successfully applied for crack detection in asphalt pavement images. The network was benchmarked against two different datasets; online and self-collected datasets and proven to produce good classification performance. The result using self-collected dataset also indicated highest accuracy of 98.80% which indicated that this proposed DCNN model is robust and more reliable compared to the previously developed methods. The proposed method can be used can be used to assist road surveyor in pavement crack detection, so that the process of detection can be done faster and more efficient. This will help in reducing cost of maintenance and enhancing safety of road users.

Acknowledgement
The authors would like to thank Advanced Rehabilitation Engineering in Diagnostic and Monitoring Research Group (AREDiM) and Universiti Teknologi MARA, Pulau Pinang for their support and assistance with this project. This research is funded by the Fundamental Research Grant Scheme (FRGS), Grant No: FRGS/1/2019/TK04/UITM/02/30. Also, appreciation goes to the Ministry of Higher Education (MOHE) for giving great privilege in providing scholarship as inspiring encouragement for the success of this research.

References
[1] H. Nhat-Duc, Q.-L. Nguyen, and V.-D. Tran, “Automatic recognition of asphalt pavement cracks using metaheuristic optimized edge detection algorithms and convolution neural network,” Automation in Construction, vol. 94, no. January, pp. 203–213, Oct. 2018.
[2] V. Mandal, L. Uong, and Y. Adu-Gyamfi, “Automated Road Crack Detection Using Deep Convolutional Neural Networks,” in 2018 IEEE International Conference on Big Data (Big Data), 2018, pp. 5212–5215.
[3] Q. Zou, Y. Cao, Q. Li, Q. Mao, and S. Wang, “CrackTree: Automatic crack detection from pavement images,” Pattern Recognition Letters, vol. 33, no. 3, pp. 227–238, Feb. 2012.
[4] D. Zhang, Q. Li, Y. Chen, M. Cao, L. He, and B. Zhang, “An efficient and reliable coarse-to-fine approach for asphalt pavement crack detection,” Image and Vision Computing, vol. 57, pp. 130–146, Jan. 2017.
[5] A. Mohan and S. Poobal, “Crack detection using image processing: A critical review and analysis,” Alexandria Engineering Journal, vol. 57, no. 2, pp. 787–798, Jun. 2018.
[6] H. Oliveira and P. L. Correia, “Automatic Road Crack Detection and Characterization,” IEEE Transactions on Intelligent Transportation Systems, vol. 14, no. 1, pp. 155–168, Mar. 2013.
[7] A. Cord and S. Chambon, “Automatic Road Defect Detection by Textural Pattern Recognition Based on AdaBoost,” Computer-Aided Civil and Infrastructure Engineering, vol. 27, no. 4, pp. 244–259, Apr. 2012.

[8] Y. Shi, L. Cui, Z. Qi, F. Meng, and Z. Chen, “Automatic Road Crack Detection Using Random Structured Forests,” IEEE Transactions on Intelligent Transportation Systems, vol. 17, no. 12, pp. 3434–3445, Dec. 2016.

[9] A. Zhang et al., “Automated Pixel-Level Pavement Crack Detection on 3D Asphalt Surfaces Using a Deep-Learning Network,” Computer-Aided Civil and Infrastructure Engineering, vol. 32, no. 10, pp. 805–819, Oct. 2017.

[10] Y.-J. Cha, W. Choi, and O. Büyükoztürk, “Deep Learning-Based Crack Damage Detection Using Convolutional Neural Networks,” Computer-Aided Civil and Infrastructure Engineering, vol. 32, no. 5, pp. 361–378, May 2017.

[11] Z. Fan, Y. Wu, J. Lu, and W. Li, “Automatic Pavement Crack Detection Based on Structured Prediction with the Convolutional Neural Network,” pp. 1–9, Feb. 2018.

[12] S. Gao, Z. Jie, Z. Pan, and F. Qin, Automatic Recognition of Pavement Crack via Convolutional Neural Network, vol. 10092. Berlin, Heidelberg: Springer Berlin Heidelberg, 2017.

[13] N. T. H. Nguyen, T. H. Le, S. Perry, and T. T. Nguyen, “Pavement Crack Detection using Convolutional Neural Network,” in Proceedings of the Ninth International Symposium on Information and Communication Technology - SoICT 2018, 2018, pp. 251–256.

[14] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” Nature, vol. 521, no. 7553, pp. 436–444, May 2015.

[15] M. Hashemi, “Enlarging smaller images before inputting into convolutional neural network: zero-padding vs. interpolation”, J Big Data 6, 98, 2019.

[16] V. Shankar, V. Kumar, U. Devagade, V. Karanth, K. Rohitaksha, Heart Disease Prediction Using CNN Algorithm, SN Computer Science (2020) 1:170, pp. 1-8, 2020.