Road Damage Detection Utilizing Convolution Neural Network and Principal Component Analysis

Elizabeth Endri[1], Alaa Sheta[2], Hamza Turabieh[3]
Computer Science Department, Southern Connecticut State University, CT 06514, USA[1,2]
Department of Information Technology, Taif University, Taif, Saudi Arabia[3]

Abstract—**Roads should always be in a reliable condition and maintained regularly. One of the problems that should be maintained well is the pavement cracks problem. This a challenging problem that faces road engineers, since maintaining roads in a stable condition is needed for both drivers and pedestrians. Many methods have been proposed to handle this problem to save time and cost. In this paper, we proposed a two-stage method to detect pavement cracks based on Principal Component Analysis (PCA) and Convolutional Neural Network (CNN) to solve this classification problem. We employed a Principal Component Analysis (PCA) method to extract the most significant features with a different number of PCA components. The proposed approach was trained using a Mendeley Asphalt Crack dataset, which contains 400 images of road cracks with a 480×480 resolution. The obtained results show how PCA helped in speeding up the learning process of CNN.**

Keywords*—*Pavement crack; Convolutional Neural Network (CNN); Principal Component Analysis (PCA)

I. Introduction

Transportation systems depend mainly on the quality of the pavement's condition. Pavement should be able to handle traffic and environmental load for many years [1]. Subsequently, roads might be damaged over time and demonstrate distresses. To guarantee long-long-term performance and an efficient level of service, they need to be well-preserved and go through a frequent maintenance operation. Semi-automated and automated imaging-based methods are employed to provide the early detection of pavement cracks [2]. In general, roads should have good features such as shape, surface, and friction to enable users to feel safe while using them. Authorized transportation agencies are responsible for maintaining roads regularly and maintain them in good condition. In general, roads should have a prearranged schedule to keep the road safe for the public [3].

The U.S. Department of Transportation (DOT) spends billions of dollars every year for building new roads and bridges. For example, in 2018, the DOT spent more than $63 Billion on major transportation infrastructure investments throughout the USA. Meanwhile, in 2017, some unfortunate claims were reported by the American Society of Civil Engineers (ASCE) Infrastructure Report, where the roads in the USA had a "D+" grade for the road infrastructure. It was pointed out that on a scale 1 out of every five miles of highway in the USA has a bad pavement infrastructure condition [4].

One of the main difficulties of maintaining road safety is pavement crack detection, which is a challenging problem that faces road engineers all year [5]. There are several causes of pavement cracks which include poor construction, bad weather conditions, and inadequate structural support for large vehicles [6]. Traditionally, cracks have been detected through a visual process that was proven to be a tedious, time-consuming, and expensive method with an especially low rate of effectiveness. Normally, a road maintenance operator needs a great deal of related knowledge and subjectivity to deal with such problems [7]. The manual inspection is also extremely dangerous for inspectors due to traffic hazards.

Drivers are at a high risk as well. Traffic accidents are a cause of serious concern for transportation engineers and researchers. Road accidents result in significant social and economic costs. Fluctuations in the number of accidents have occurred on highways each year[8]. Some life-threatening consequences of pavement deterioration and defects are skidding, accidental driving off-road, and spontaneous maneuvering to eliminate road infrastructure problems [9] which places the driver and others at high-risk. Besides, poor surface macrotexture and microtexture could lead to hydro-planning and inconsistent tire pavement contact resulting in the reduction of tires gripping the pavement which can cause accidents [10]. Roads or pavement engineers usually inspect all types of cracks, distress, and unevenness routine manner by gathering road condition data. Gathering road data should be implemented in all weather and traffic conditions. This process may suffer from some human errors and consume time [11]. Therefore, it is important to have well-defined strategies for monitoring and maintaining roads [12].

The motivation behind this work can be described as a response to the thousands of needless deaths each year that occur due to pavement distresses all around the world. Maintenance workers put their lives on the line to perform manual inspections of roads. According to the United States Department of Labor Occupational Safety and Health Administration, out of 4,674 worker fatalities in private industry in 2017, 20.7% were in construction. In other words, one in five worker deaths last year were in construction. As a result, this motivates us to investigate the performance of the CNN method with PCA as a feature selection to detect pavement cracks inside images.

The objective of this work is to develop an intelligent approach based on CNN for road damage detection to achieve a trustworthy detection and classification of cracks from obtained 2D concrete and asphalt pavement images. This paper is organized as follows. Section II presents a literature review of the pavement crack detection research. In Section III, we discuss the PCA and CNN elements of the proposed method. A description of the dataset and the way we split the training and testing data is presented in Section VI. The experimental results based on a well-known pavement crack dataset is presented in Section VII. Finally, concluding remarks and future works are presented in Section VIII.

II. Literature Review

In the past, many research papers investigate the pavement crack problem as an image processing problem. For example, Sy *et al.* [13] applied three operations (i.e., bi-level threshold, morphological, and projection) to detect pavement cracks. The experimental data was on three kinds of images: laboratory images, static images, and AMAC reg images. Li *et al.* [14] studied this problem by proposing an approach as a thresholding method based on neighboring differential histogram statistics. Oliveira and Correia [15] handle pavement crack problems inside images by proposing a local thresholding approach based on non-overlapping blocks.

Recently, the deep convolutional neural network has been proven to show great advantages in image classification and an excellent classifier for pavement cracks. In [7], the authors extracted small patches from cracked pavement images as inputs to generate a large training database. Their proposed CNN network included 4 convolutional layers with 2 max-pooling layers and 3 fully connected layers. The proposed method had an accuracy of 91% and a recall rate of 91%. In the CrackIT project published in [16], the author used the mean and standard deviation for the unsupervised learning algorithm to distinguish blocks with cracks against blocks without cracks. They assigned severity levels to identify crack segments which relied on the computed measurement of the crack's width. The ratio between the crack segment area and the number of crack pixels belonging to the crack skeleton was computed. The results showed an accuracy of 97%, a recall score of 98.4%, and a precision of 95.5%. The drawback in their method was that they were dealing with extremely thin cracks (many of which were less than 2 mm), which proved to be a difficult task.

In [17], AlexNet created by Alex Krizhevsky used Rectified Linear Units (ReLU) instead of the $tanh$ function, which was standard at the time. ReLU's advantage over other methods is in its training time. CNN used a ReLU layer to provide a 25% error on the CIFAR-10 dataset which was six times faster than a CNN using the $tanh$ function.

In separate work, Honyan Xu *et al.* proposed an end to end crack detection based on the CNN with 28 layers, including 16 convolutional layers and earned a 90.19% test accuracy [18]. The Unmanned Aerial Vehicles (UAVs) was also used for crack inspection and monitoring. In [19], the authors proposed to simulate the pre-trained deep learning models with transfer learning methods to detect the pavement cracks based on UAV images of civil infrastructure. They employed small and complex UAV images for training and validation phases. The obtained results show that the accuracy of the proposed methods was 90% in finding cracks in practical situations with no need for augmentation and pre-processing.

In [20], the author introduced a CNN model structure to solve the crack detection and classification problems. They used a digital camera to collect images of various resolutions (i.e., 32 × 32 and 64 × 64). Two CNN networks were trained based on image resolution to detect if there was a crack or not. To achieve the second goal, the authors converted the image to binary ones with two types of crack, transverse, and longitudinal. The output from the first stage was feed to a second CNN to classify the type of crack. The finding was interesting since the images with low resolutions provided a higher classification accuracy. For 32 × 32 resolution images, the recall, precision, and accuracy calculated was 98.0%, 99.4%, and 99.2% respectively for crack and non-crack detection, while the performance for classification (i.e., transverse and longitudinal) reached the accuracy of 98% and 97%.

III. Methods

Developing an intelligent and trustworthy detection model based on CNN to detect cracks inside 2D concrete and asphalt pavement images is the main objective of this paper. The adopted database of concrete and asphalt pavement has images obtained by a 2D area digital scanning method. In this section, we shall describe the adopted methodology to solve the pavement crack detection problem.

A. Preprocessing: PCA

Principal Component Analysis commonly referred to as PCA, is a linear transformation of data. It is one of the most widely used methods of re-framing the data given [21], [22]. It measures the distances from the data to the line and tries to find the line that minimizes those distances or it can try to find the line that maximizes the distances from the projected points to the origin. It is a data transformation technique that can make it easier to use with reduction later. Data must be standardized. Dimensions will be centered around zero and have a standard deviation of 1. PCA will find a new axis, or a new attribute such that the data is maximized.

PCA works as a dimension reduction and data analysis tool. PCA has been applied successfully in a vast research area such as data mining, image processing, and artificial intelligence [23], [24]. PCA is one of the most well-known methods of factor analysis to project high-dimensional data (e.g., images) into low dimensional data based on a linear transformation without losing the value of original features [25]. So, the PCA method will reduce the number of variables and group these new variables into groups called *factors*, which improve the overall performance of machine learning classifiers such as execution time and memory usage.

The basic idea of PCA appeared in 1901 by Karl Pearson [26]. In 1936, Harold Hotelling [27] improved and developed the classical PCA. PCA is a method that aims in simplifying a multidimensional dataset to lower dimensions for analysis and visualization. In general, PCA works by converting the correlated feature variables into a new set of linearly uncorrelated features variables, which is called principal components. The main condition of PCA is that the number of PCA components should be less than or equal to the number of original features variables.

In this paper, we employed the PCA as a pre-processing step of the images before sending it to CNN to reduce the data size and improve the overall performance of CNN. Given a set of pavement crack image $\{x^1, x^2, x^3, ..., x^n\}$, the PCA works as follows:

- First: We calculate the covariance matrix of x and $\sum x$, using Equation 1.

$$\sum x = \frac{1}{m}\sum_{i=0}^{m}(x^i)(x^i)^T \qquad (1)$$

- Second: We compute the eigenvectors of $\sum x$, and construct a matrix as shown in Equation 2, where u_1 represents the first eigenvector, u_2 represents the second eigenvector, and so on. Equation 3 shows the calculation that is used to construct the input features maps that are uncorrelated with each other.

  The covariance matrix for x_{rot} can be extracted from the diagonal matrix from U, whose diagonal elements λ_1, λ_2, λ_3,..., λ_n. Where λ_i presents corresponding eigenvalues of eigen vector matrix U.

- Finally: PCA is evaluated based on Equation 4.

$$U = \begin{bmatrix} | & | & & | \\ u_1 & u_2 & \dots & u_n \\ | & | & & | \end{bmatrix} \qquad (2)$$

$$x^i_{rot} = U^T x^i \qquad (3)$$

$$x_{PCA,i} = \frac{x_{rot,i}}{\sqrt{\lambda_i}} \qquad (4)$$

B. What is ANN?

Artificial Neural Networks (ANNs) are computational processing systems that were inspired by how biological nervous systems function [28]. In reality, the neural system is a very complex one that consists of an extremely large number of neurons. Each neuron is designed to receive an input signal(s) from its dendrites and generate an output signal(s). the output signal(s) goes through the axon, which transfers the generated signal to the next neuron using synapses. Once a set of input signals reaches a predetermined threshold value, the neuron is triggered, which simulates the real functions inside the human brain (see Fig. 1 [29]).

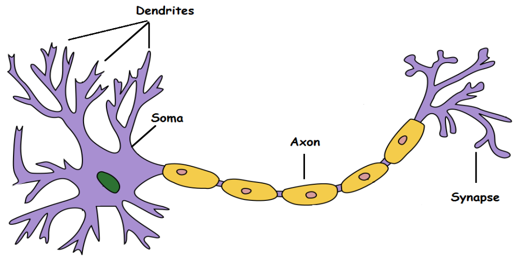


Fig. 1. A Biological Neuron [29].

ANNs consist of interconnected nodes, called neurons, that learn from given input to optimize to final output. These artificial neurons have numeric weights attached to them. These weights will be optimized through the training phase. The performance of well-trained ANN will show high performance with a piece of datum or pattern to recognize or identify (see Fig. 2). Using a suitable learning algorithm, these units are efficient in generating a function that maps a relationship between inputs and output training examples.

ANN uses a training dataset (i.e., images, row data, etc.) as input data. The input layer handles the training dataset, which is connected to the next layer (i.e., the hidden layer). The hidden layer will manipulate the data and tune the connection weights before sending it to the output layer. Each ANN should have a learning algorithm (e.g., BackPropagation, Convolutional Neural Network, Long Short-Term Memory, etc.) that tunes the ANN weights to enhance the overall performance of ANN by reducing the error between the real output (i.e., actual) and obtained output (i.e., predicted) from ANN [30], [31], [32].

$$S = \sum_{i=0}^{n} w_i x_i \qquad (5)$$

$$\phi(S) = \frac{1}{1+e^{-S}} \qquad (6)$$

Several tuning parameters should be designated before we can use ANN to be trained. They include the number of layers in the hidden layer, the type of sigmoid function for the neurons, and the adopted learning algorithm (see Fig. 2 [33])

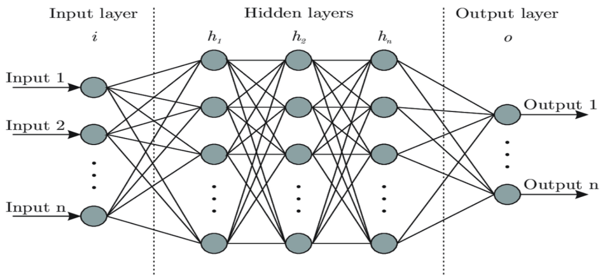


Fig. 2. Fully Connected ANN Architecture [33].

IV. What is CNN?

The earliest CNN model called the leNet-5 model was proposed by LeCun in 1998 [34]. CNN can be thought of as a close family member of the traditional ANN. The main structure of CNN is motivated by the discovery of the visual cortex in the brain, which contains a large number of cells that detect the light in the small receptive fields, and overlapping sub-regions of the visual field (see Fig. 3). These cells act as local filters over the input space, and the more complex cells have larger receptive fields.

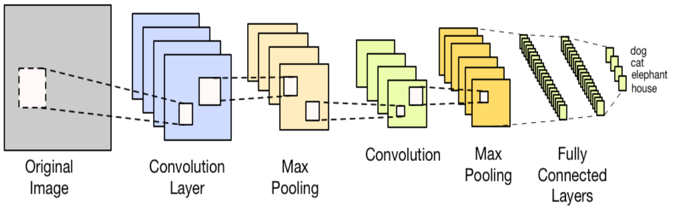


Fig. 3. An Overview of a CNN model.

Therefore, a simple CNN is a series of layers, and every layer of a CNN converts one volume of activations to another through a differentiable function. We use three main types of layers to build CNN architectures: Convolutional Layer, Pooling Layer, and Fully-Connected Layer (exactly as seen in regular Neural Networks). We will stack these layers to form a full CNN architecture.

A. Convolution Layer

Traditional neural networks are fully connected in every layer, while convolutional layers in CNNs use the convolutional operation [35]. The convolution layer in CNN operates the function that is performed by the cells in the visual cortex. The neurons in CNNs are self-optimize through learning. Each neuron receives input and operates. CNN's have an input layer, various hidden layers, and an output layer. These hidden layers use a mathematical model to pass on results to the following layer.

Convolution is the first layer used to extract features from an input image and preserves the relationship between pixels by learning image features using small tiles of input data [1], [36]. Essentially, the convolutional layer is a mathematical equation that takes two inputs such as an image matrix and a filter or kernel. Convolution uses a small square matrix, which preserves the spatial relationships among pixels, to learn image features [37]. The convolution layer can do quite a few operations with different filters including edge detection, blurring, and sharpening an image [38].

The convolution layer is the essential component of a convolutional neural network. The convolution layer includes of a set of independent filters. Each filter is individually convolved with the image, and feature maps are obtained. In general, if we convolve an image of size $N \times M$ with a filter of size $l \times k$, we get an output feature map of size $O_{width} \times O_{height}$ as given in Equation 7.

$$\begin{aligned} O_{width} &= \frac{N - l + 2p_l}{s_l} + 1 \\ O_{height} &= \frac{M - k + 2p_k}{s_k} + 1 \end{aligned} \tag{7}$$

where p_l and p_k are the padding in both width and height, respectively, and s_l and s_k represent the stride in both horizontal and vertical directions. Thus, if we apply a convolution operation with a filter of size 5 × 5 on an image with a size 32 × 32, the result will be a feature map of size 28 × 28, with a zero-padding and stride of one. The output feature map is acquired by the convolution of the input maps with a linear filter, adding a bias term and then applying a nonlinear function. The output can be generally denoted by the formula as in Equation 8

$$X_j^q = f(\sum_{I \in I_j} X_i^{q-1} \times W_{ij}^q + b_j^q) \tag{8}$$

where q represents the layer number, W_{ij} represents the convolutional kernel, b_j represents bias, I_j represents the set of input maps and $f(.)$ represents the activation function. Fig. 4 shows the output features collected from the third layer after doing the feature extraction.

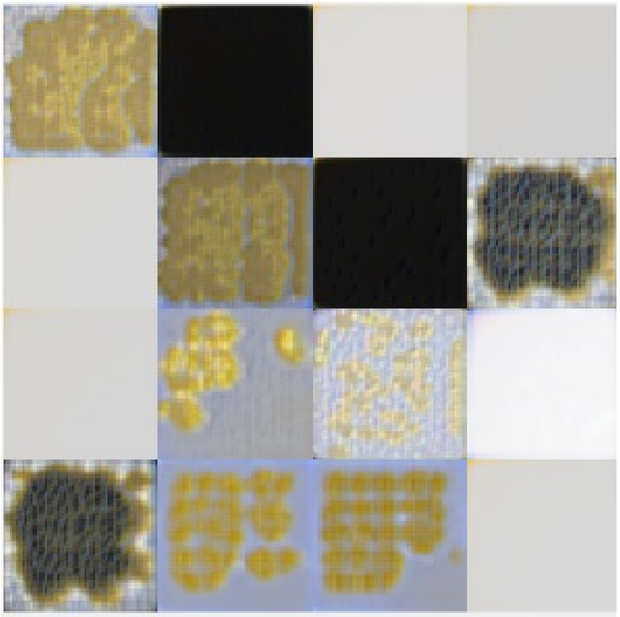


Fig. 4. The Output of the Third Convolution Layer.

B. Polling Layer

The main objective of the pooling layer is to decrease the spatial size of the representation, which will enhance the overall performance of the neural network, the number of CNN parameters, and reduce the probability of overfitting. In general, the pooling layer is located between successive convolutional layers on CNN. The pooling layer operates on sliding a two-dimensional filter over each channel of feature map and summarizing the features lying within the region covered by the filter.

The pooling layers in the CNN are used to reduce the number of parameters when the images are too large (see Fig. 5). Spatial pooling reduces the dimensionality of each map but keeps important information. It also can control over-fitting. There are two types of pooling 1) max and 2) average pooling. Max pooling is defined as a sample-based discretization process. The advantage of Max-Pooling is a massive edge detection-based matrix multiplication.

$$\begin{array}{|c|c|c|c|} \hline 12 & 20 & 30 & 0 \\ \hline 8 & 12 & 2 & 0 \\ \hline 34 & 70 & 37 & 4 \\ \hline 112 & 100 & 25 & 12 \\ \hline \end{array} \xrightarrow{2 \times 2 \text{ Max-Pool}} \begin{array}{|c|c|} \hline 20 & 30 \\ \hline 112 & 37 \\ \hline \end{array}$$

Fig. 5. An Example of a Max Pooling Operation.

C. Rectified Linear Unit

The activation function is a very important element for the CNN design such that it can learn and perform more complex tasks. Activation functions are a nonlinear function utilized to the input. Several frequently used activation functions in the literature are sigmoid, logistic activation function, tanh, and hyperbolic tangent activation function. In this work, we are adopting a ReLU activation function. ReLU stands for the Rectified Linear Unit for a nonlinear operation. The rectified linear activation function is defined as a piecewise linear function. This function can produce either the same function input if the input is position and zero if the input is negative. Fig. 6 shows the ReLU function. Equations 9 and 10 demonstrate the computations of ReLU.

$$R(z) = max(0, z) \tag{9}$$

$$R(z) = \begin{cases} z & \text{if } z \leq 0, \\ z & \text{if } 0 < 0. \end{cases} \tag{10}$$

ReLU helps to backpropagate the errors and have multiple layers of neurons being triggered by the ReLU function. ReLU helps to overcome the vanishing gradient problem and allows models to learn faster. ReLU is widely recommended to use in CNN classification models [39].

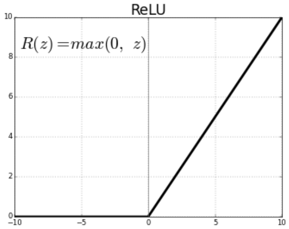


Fig. 6. Rectified Linear Unit (ReLU) Activation Function.

D. SoftMax Unit

SoftMax is another activation function like sigmoid, tanh, and ReLU. They are commonly used for the neurons in the output of the fully connected. It's defined as:

$$\sigma(z_i) = \frac{e_i^z}{\sum_{j=1}^{C} e_j^z} \tag{11}$$

where C is the number of classes, z is the input vector, and $\sigma(z_i)$ is the output class probability. SoftMax function provides a discrete probability distribution over all the given classes. The SoftMax function output is a probabilities $p_i \in [0, 1]$. The sum of the probability of all classes C is $\sum p_i = 1$.

E. Fully-Connected Layer

The fully connected (FC) means that every single neuron in the preceding layer is connected to every single neuron on the current layer. Each neuron shall have a summation followed by an activation function. The final layer of CNN is a FC layer that has a FC to all activation functions in the previous layer, as observed in the traditional ANN. These activation functions are used to compute the CNN final output via a matrix multiplication followed by a bias offset. In the CNN, the FC layer merges all the features obtained from the previous convolutional and sub-sampling layers.

F. Evaluation Metrics

The evaluation metrics that will be used in this paper are accuracy, precision, and recall. These metrics were chosen because they are commonly used in classification problems. They are defined as the following:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{12}$$

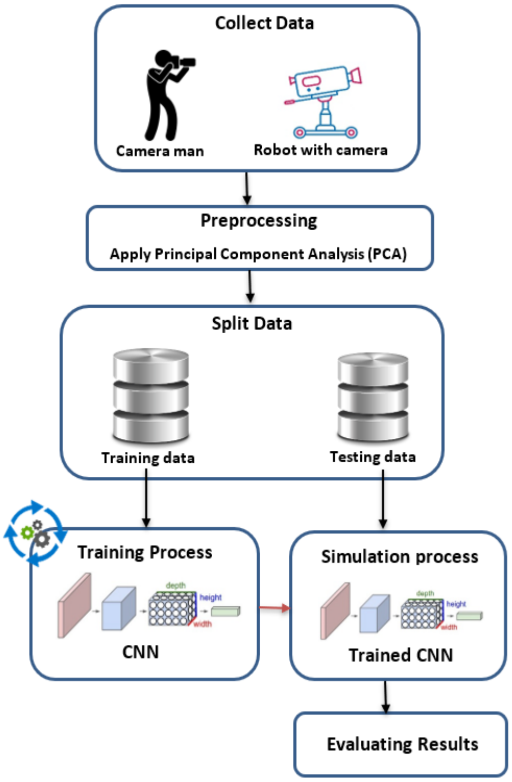


Fig. 7. Proposed CNN based-Detection Method.

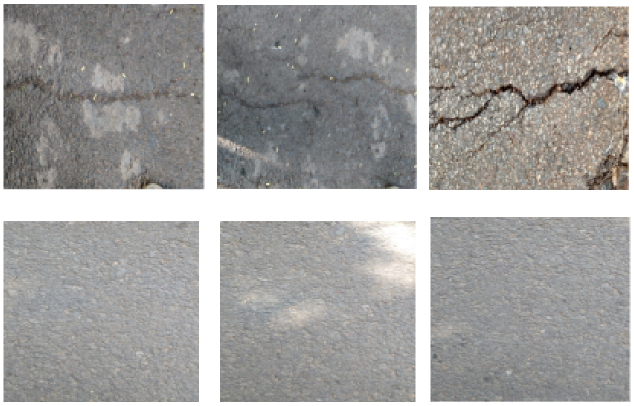


Fig. 8. Mendeley Asphalt Crack Dataset. Upper Row: Image Cracks, Lower Row: No Crack.

$$Precision = \frac{TP}{TP + FP} \tag{13}$$

$$Recall = \frac{TP}{TP + FN} \tag{14}$$

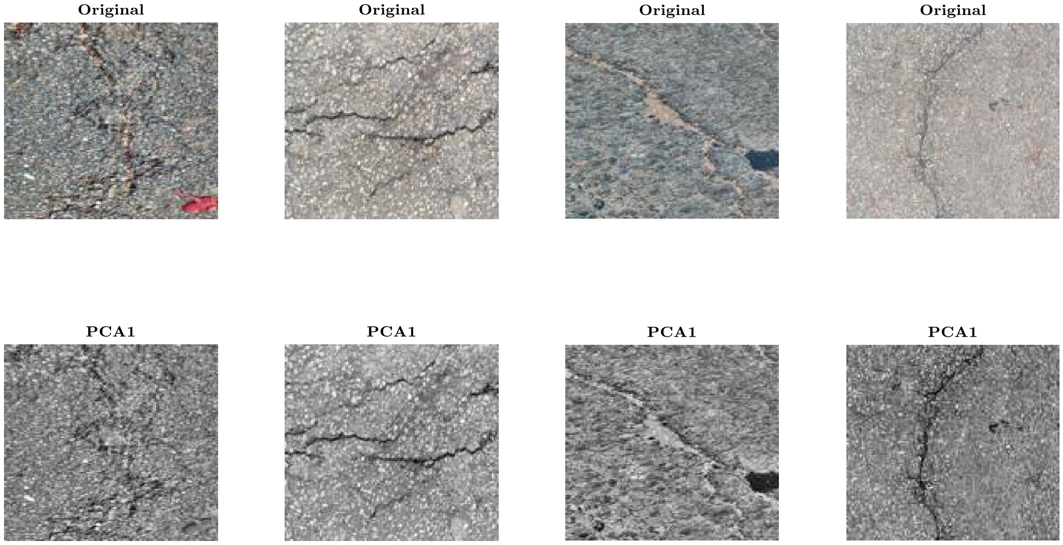
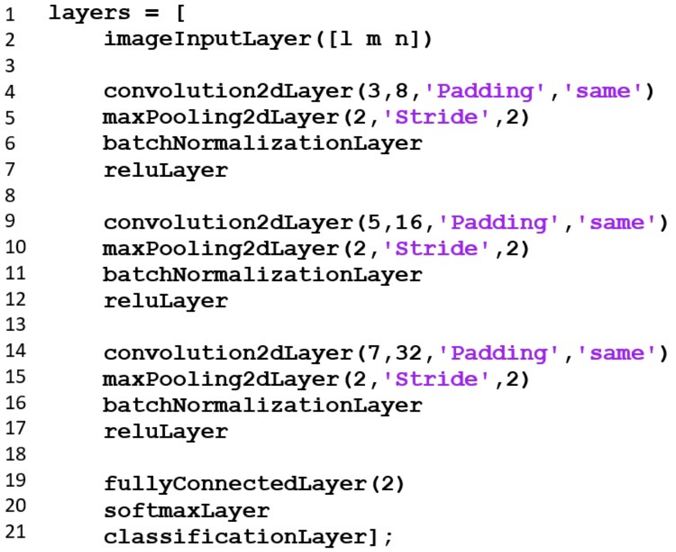


Fig. 9. First Row: Sample Images and Second Row: PCA-Crack Images.



```
layers = [
    imageInputLayer([l m n])

    convolution2dLayer(3,8,'Padding','same')
    maxPooling2dLayer(2,'Stride',2)
    batchNormalizationLayer
    reluLayer

    convolution2dLayer(5,16,'Padding','same')
    maxPooling2dLayer(2,'Stride',2)
    batchNormalizationLayer
    reluLayer

    convolution2dLayer(7,32,'Padding','same')
    maxPooling2dLayer(2,'Stride',2)
    batchNormalizationLayer
    reluLayer

    fullyConnectedLayer(2)
    softmaxLayer
    classificationLayer];
```

Fig. 10. Layer Structure for CNN Model.

TABLE I. OBTAINED RESULTS FOR TRAINING DATASET.

	PCA Component	Best	Worst	Avg.	Std.
Without PCA	—	97.78	85.02	93.13	2.92
With PCA	2	98.06	91.62	95.60	1.98
	5	95.08	90.11	93.18	1.51
	10	97.67	94.34	**96.62**	0.93
	15	98.13	93.75	96.03	1.11

where TP, FP, TN and FN are the true positive, false positive, true negative, and false-negative, respectively.

V. PROPOSED CNN-BASED METHOD

The proposed method used in this work is depicted in Fig. 7. The proposed method is a combination of PCA and CNN methods. In the first step, we collect data for pavement cracks using a traditional method (i.e., cameraman) or an intelligent method (i.e., a robot with a camera). After collecting data, it is important to analyze the most valuable features by extracting image features using the PCA method. The proposed method will enhance the performance of CNN convergence.

VI. DATASET

A pavement crack dataset called the Asphalt Crack dataset is used in this work. The data consists of 400 images. It is a contribution by Jayanth Balaji, Thiru Balaji, Dinesh M S, Binoy Nair, and Harish Ram D.S. The dataset was published on April 26, 2019 [40]. Fig. 8 depicts samples of the dataset (i.e., cracked and not cracked pavements).

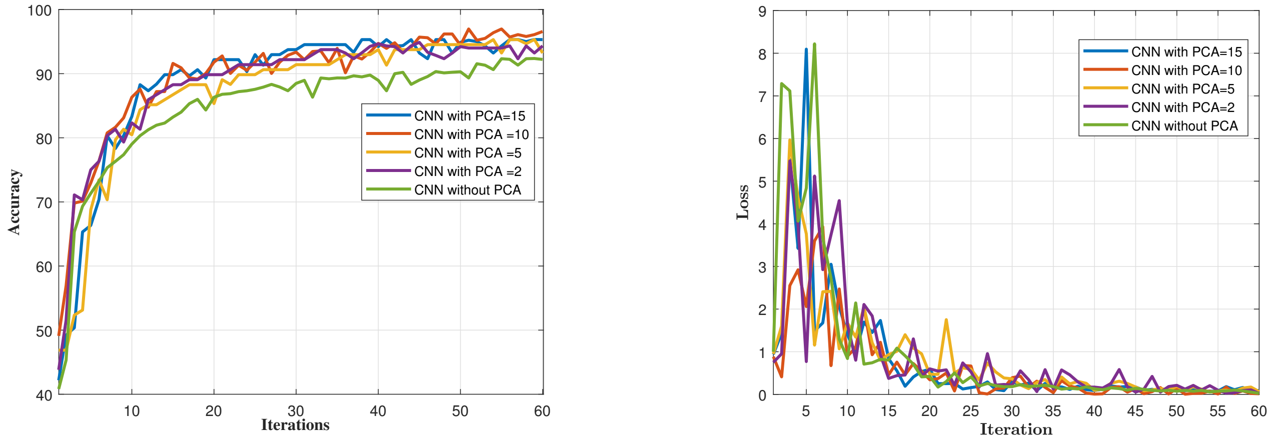
A. Training and Testing Data

The hold-out method is the simplest kind of cross-validation method. The main basic idea of the hold-out method is to divide them into two groups: Training and Test/Validation dataset. The training dataset is used to train the model (i.e., CNN) and evaluate the trained model using the Test/Validation dataset. This approach of splitting data is applicable to image processing applications. We adopted the classical holdout method for splitting data.

VII. EXPERIMENTAL RESULT

The proposed method for pavement crack detection was simulated on an Intel Core i7-7700HQ 2.8-GHz processor with 16 GB RAM and implemented using MATLAB R2019b environment [41]. First, we preprocessed the crack images using PCA. We created a set of feature images from the original crack images data set as given in Fig. 9. These are the first features created from PCA. These images were used as input to the CNN for further processing.

A sample code that shows the CNN architecture is presented in Fig. 10. In this work, we employed two types of experiments: (i) without PCA, and (ii) with PCA. Moreover, we explored the performance of PCA with several numbers



(a) Average Convergence Curves.

(b) Loss Convergence Curves.

Fig. 11. Experimental Results showing Convergence Curves.

of PCA component (i.e., 2, 5, 10, and 15). For each type of experiment, we executed our program eleven times. Table I shows the obtained results for the training dataset. The performance of CNN with PCA_{10} (i.e., PCA with ten components) outperforms all other methods based on average accuracy of 96.62 and standard deviation of 0.93. Fig. 11a explores the average convergence curves of the accuracy during the training process for all trained models. The performance of CNN is improved after employing the PCA. Fig. 11b demonstrates the performance of the CNN models with and without PCA. The performance of CNN with PCA is improved and able to detect the pavement cracks robustly compared to CNN without PCA.

In Fig. 12a we show the CNN performance based on eleven runs. It was found that the performance results of CNN without PCA is not the best. Table II shows the performance of the proposed method over the testing dataset. Again it is clear that the performance of CNN with PCA_{10} outperforms all other models based on average and standard deviation. Fig. 12b shows the performance of all trained models over the testing dataset. The PCA can enhance the performance of CNN. Moreover, the performance of CNN with PCA_{10} is the most suitable method for the pavement crack detection method.

TABLE II. Obtained results for testing dataset.

	PCA Component	**Best**	**Worst**	**Avg.**	**Std.**
Without PCA	—	91.36	84.36	88.61	2.35
With PCA	2	94.36	86.23	90.44	2.48
	5	94.39	87.42	91.97	2.04
	10	95.70	91.37	**94.01**	1.13
	15	95.31	88.23	91.55	1.92

Tables III and IV report the obtained average results (i.e., Accuracy, Precision, and Recall) for training and testing datasets, respectively. The performance of CNN with PCA component equals 10 outperforms all other models based on the average results for 11 independent runs.

TABLE III. Statistical analysis for all models for Training dataset.

	PCA Component	Accuracy	Precision	Recall
Without PCA	—	93.13	0.92	0.97
With PCA	2	95.60	0.91	0.93
	5	93.18	0.93	0.96
	10	**96.62**	**0.96**	**0.98**
	15	96.03	0.94	0.97

TABLE IV. Statistical analysis for all models for testing dataset.

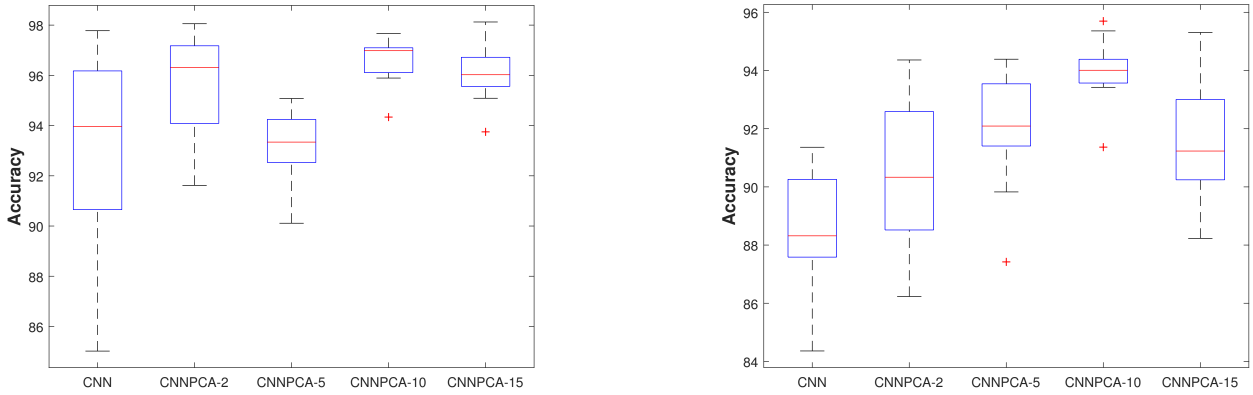
	PCA Component	Accuracy	Precision	Recall
Without PCA	—	88.61	0.82	0.84
With PCA	2	90.44	0.93	0.97
	5	91.97	0.89	0.90
	10	**94.01**	**0.94**	**0.93**
	15	91.55	0.87	0.84

To examine the obtained results and show how the number of PCA components affects the classifiers' performance, we employed a statistical analysis and comparison based on the Wilcoxon statistical test with a significance level of 0.05. Table V shows the p-values of the obtained results between a different number of PCA components. All the obtained p-values are less than 0.05, which means that there is a statistical difference between them. For example, the p-value between PCA_2 and PCA_{10} is 0.040, which means the performance of CNN is not similar.

TABLE V. P-value results based on Wilcoxon test.

	P-value
PCA_2 vs. PCA_5	0.031
PCA_2 vs. PCA_{10}	0.040
PCA_2 vs. PCA_{15}	0.027
PCA_5 vs. PCA_{10}	0.035
PCA_5 vs. PCA_{15}	0.049
PCA_{10} vs. PCA_{15}	0.039

Finally, from the obtained results, we can conclude that PCA as a feature extraction can enhance the performance of CNN. Moreover, the proposed approach can examine a huge



(a) Box-Plot for the Training Dataset.

(b) Box-Plot for the Testing Dataset.

Fig. 12. Experimental Results for the Training and Testing Dataset.

number of images automatically, which save time, cost for pavement cracks detection, and reduce risks for roads and pavements engineers.

VIII. Conclusions and Future Work

In this work, we proposed a CNN-based method to automate the pavement crack detection process. The main idea of the proposed method is to combine CNN with PCA to speed up the learning process. CNN was employed as a classification method, while PCA as a feature extraction one. We examined the performance of our proposed method on a public dataset that contains 400 images. We also explored several numbers of PCA components (i.e., 2, 5, 10, and 15). The obtained results show that CNN with PCA_{10} outperforms all other models. In future work, we will examine different parameters setting and employed an optimization algorithm such as a genetic algorithm, to optimize the PCA parameters.

Acknowledgment

The authors would like to thank the Computer Science Department, Southern Connecticut State University for the financial support of this research work.

References

[1] S. Chambon and J.-M. Moliard, "Automatic road pavement assessment with image processing: Review and comparison," *International Journal of Geophysics*, vol. 2011, 03 2011.

[2] A. Cubero-Fernandez, F. J. Rodríguez Lozano, R. Villatoro, J. Olivares, and J. Palomares, "Efficient pavement crack detection and classification," *Eurasip Journal on Image and Video Processing*, vol. 2017, 12 2017.

[3] F. Yang, L. Zhang, S. Yu, D. Prokhorov, X. Mei, and H. Ling, "Feature pyramid and hierarchical boosting network for pavement crack detection," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–11, 2019.

[4] ASCE, "2017 infrastructure report card: Roads. american society of civil engineers (asce)," https://www.infrastructurereportcard.org/wp-content/uploads/2017/01/Roads-Final.pdf, May 2017.

[5] A. Sheta, H. Turabieh, S. Aljahdali, and A. Alangari, "Pavement crack detection using a lightweight convolutional neural network," *Proceedings of 35th International Confer*, vol. 69, pp. 214–223, 2020.

[6] A. K. Schindler and B. F. McCullough, "Importance of concrete temperature control during concrete pavement construction in hot weather conditions," *Transportation Research Record*, vol. 1813, no. 1, pp. 3–10, 2002.

[7] Z. Fan, Y. Wu, J. Lu, and W. Li, "Automatic pavement crack detection based on structured prediction with the convolutional neural network," *arXiv preprint arXiv:1802.02208*, 2018.

[8] S. N. Baskara, H. Yaacob, M. R. Hainin, and S. A. Hassan, "Accident due to pavement condition–a review," *Jurnal Teknologi*, vol. 78, no. 7-2, 2016.

[9] M. Ergun, S. Iyinam, and A. F. Iyinam, "Prediction of road surface friction coefficient using only macro-and microtexture measurements," *Journal of transportation engineering*, vol. 131, no. 4, pp. 311–319, 2005.

[10] L. F. Beck, A. M. Dellinger, and M. E. O'neil, "Motor vehicle crash injury rates by mode of travel, united states: using exposure-based methods to quantify differences," *American Journal of Epidemiology*, vol. 166, no. 2, pp. 212–218, 2007.

[11] K. Gopalakrishnan, S. K. Khaitan, A. Choudhary, and A. Agrawal, "Deep convolutional neural networks with transfer learning for computer vision-based data-driven pavement distress detection," *Construction and Building Materials*, vol. 157, pp. 322 – 330, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S09500618173193355

[12] S. Saat, A. R. M. Kamil, M. Z. M. Tumari, and A. S. R. A. Subki, "Development of an autonomous robot for inspection system," in *2018 IEEE 14th International Colloquium on Signal Processing Its Applications (CSPA)*, March 2018, pp. 272–276.

[13] N. T. Sy, M. Avila, S. Begot, and J. C. Bardet, "Detection of defects in road surface by a vision system," in *MELECON 2008 - The 14th IEEE Mediterranean Electrotechnical Conference*, May 2008, pp. 847–851.

[14] Q. Li and X. Liu, "Novel approach to pavement image segmentation based on neighboring difference histogram method," in *2008 Congress on Image and Signal Processing*, vol. 2, May 2008, pp. 792–796.

[15] H. Oliveira and P. L. Correia, "Automatic road crack segmentation using entropy and image dynamic thresholding," in *2009 17th European Signal Processing Conference*, Aug 2009, pp. 622–626.

[16] H. Oliveira and P. L. Correia, "Crackit an image processing toolbox for crack detection and characterization," *2014 IEEE International Conference on Image Processing (ICIP)*, 2014.

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, ser. NIPS'12. USA: Curran Associates Inc., 2012, pp. 1097–1105.

[18] H. Xu, X. Su, Y. Wang, H. Cai, K. Cui, and X. Chen, "Automatic bridge crack detection using a convolutional neural network," *Applied Sciences*, vol. 9, no. 14, p. 2867, 2019.

[19] K. Gopalakrishnan, H. Gholami, A. Vidyadharan, Alok, Choudhary, and A. Agrawal, "Crack detection in unmanned aerial vehicle images of civil infrastructure using pre-trained deep learning model," vol. 8, no. 1, 2018, p. 1—14.

[20] N. A. M. Yusof, M. K. Osman, M. H. M. Noor, A. Ibrahim, N. M. Tahir, and N. M. Yusof, "Crack detection and classification in asphalt pavement images using deep convolution neural network," in *2018 8th IEEE International Conference on Control System, Computing and Engineering (ICCSCE)*, 2018, pp. 227–232.

[21] I. Jolliffe, *Principal Component Analysis*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 1094–1096.

[22] I. Jolliffe and J. Cadima, "Principal component analysis: A review and recent developments," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 374, p. 20150202, 04 2016.

[23] S. C. Ng, "Principal component analysis to reduce dimension on digital image," *Procedia Computer Science*, vol. 111, pp. 113–119, 12 2017.

[24] S. Yu, K. Yu, V. Tresp, H.-P. Kriegel, and M. Wu, "Supervised probabilistic principal component analysis," in *KDD 2006*, Max-Planck-Gesellschaft. New York, NY, USA: ACM Press, Aug. 2006, pp. 464–473.

[25] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometrics and Intelligent Laboratory Systems*, vol. 2, no. 1, pp. 37 – 52, 1987.

[26] K. Pearson, "Liii. on lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 559–572, 1901.

[27] H. Hoteeling, "Relations Between Two Sets of Variates," *Biometrika*, vol. 28, no. 3-4, pp. 321–377, 12 1936. [Online]. Available: https://doi.org/10.1093/biomet/28.3-4.321

[28] K.-l. Hsu, H. V. Gupta, and S. Sorooshian, "Artificial neural network modeling of the rainfall-runoff process," *Water resources research*, vol. 31, no. 10, pp. 2517–2530, 1995.

[29] W. Mcculloch and W. Pitts, "A logical calculus of ideas immanent in nervous activity," *Bulletin of Mathematical Biophysics*, vol. 5, pp. 127–147, 1943.

[30] A. Nigrin, *Neural networks for pattern recognition*, ser. A Bradford book. Cambridge, Mass, London: The MIT Press, 1993. [Online]. Available: http://opac.inria.fr/record=b1126290

[31] J. Leonard and M. A. Kramer, "Improvement of the backpropagation algorithm for training neural networks," *Computer Chemical Engineering*, vol. 14, pp. 337–343, 1990.

[32] A. K. Jain, J. Mao, and K. Mohiuddin, "Artificial neural networks: A tutorial," *IEEE Computer*, vol. 29, pp. 31–44, 1996.

[33] F. Bre, J. Gimenez, and V. Fachinotti, "Prediction of wind pressure coefficients on building surfaces using artificial neural networks," *Energy and Buildings*, vol. 158, 11 2017.

[34] W.-q. Huang and Q. Wu, *Image Retrieval Algorithm based on Convolutional Neural Network: Selected Papers from CSMA2016*, 12 2017.

[35] H. D. Beale, H. B. Demuth, and M. Hagan, "Neural network design," *Pws, Boston*, 1996.

[36] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105.

[37] A. de Bruijn, V. Muhonen, T. Albinonistraat, W. Fokkink, P. Bloem, and B. Analytics, "Detecting offensive language using transfer learning," 2019.

[38] H. H. Aghdam and E. J. Heravi, *Guide to Convolutional Neural Networks: A Practical Application to Traffic-Sign Detection and Classification*, 1st ed. Springer Publishing Company, Incorporated, 2017.

[39] M. Bhurtel, J. Shrestha, N. Lama, S. Bhattarai, A. Uprety, and M. Guragain, "Deep learning based seed quality tester," 11 2019.

[40] J. B. A, "Asphalt crack dataset," Apr 2019. [Online]. Available: http://dx.doi.org/10.17632/xnzhj3x8v4.2

[41] MATLAB, *(R2019b)*. Natick, Massachusetts: The MathWorks Inc., 2019.