ROAD CRACK DETECTION USING DEEP CONVOLUTIONAL NEURAL NETWORK

Lei Zhang, Fan Yang, Yimin Daniel Zhang, and Ying Julie Zhu

Department of Electrical and Computer Engineering, Temple University, Philadelphia, PA 19122, USA

ABSTRACT

Automatic detection of pavement cracks is an important task in transportation maintenance for driving safety assurance. However, it remains a challenging task due to the intensity inhomogeneity of cracks and complexity of the background, e.g., the low contrast with surrounding pavement and possible shadows with similar intensity. Inspired by recent success on applying deep learning to computer vision and medical problems, a deep-learning based method for crack detection is proposed in this paper. A supervised deep convolutional neural network is trained to classify each image patch in the collected images. Quantitative evaluation conducted on a data set of 500 images of size 3264 $\times$ 2448, collected by a low-cost smart phone, demonstrates that the learned deep features with the proposed deep learning framework provide superior crack detection performance when compared with features extracted with existing hand-craft methods.

Index Terms— Deep learning, convolution neural networks, road crack detection, road survey

1. INTRODUCTION

Keeping roads in a good condition is vital to safe driving and is an important task of both state and local transportation maintenance departments. One important component of this task is to monitor the degradation of road conditions, which is labor intensive and requires domain expertise. Recently, computer vision and machine learning techniques have been successfully applied to automate road surface survey [1–5]. In this work, we focus on detecting cracks on the pavement surface, because they represent the most prevalent type of road damage and exhibit strong texture cues. A large number of recent literature in crack detection and characterization of pavement surface distresses clearly demonstrates an increasing interest in this research area [3, 4, 6, 7].

The traditional framework for crack detection designs a variety of gradient features for each image pixel, which are followed by a binary classifier to determine whether an image pixel contains a crack or not. A local binary patterns (LBP) based algorithm for crack detection is developed in [8], whereas a crack detection method using the Gabor filter is proposed in [3]. In [4], an automatic crack detection based on the tree structure, referred to as *CrackTree*, is introduced. A fully integrated system for crack detection and characterization is proposed in [9] and a comprehensive set of image processing algorithms for detection and characterization of road pavement surface crack distresses is introduced in [5]. Although hand-crafted features are widely used and support top-ranking algorithms on the well acquired data set [4, 5, 10], it is important to note that they are not discriminative enough to differentiate the crack and complex background in low level image cues.

On the other hand, the impressive performances for many medical imaging and computer vision tasks have evidently showcased the effectiveness of deep features learned by deep neural networks [11–16] which are likely to replace the conventional hand-crafted features [17]. Restricted Boltzmann machine (RBM), autoencoder and their variants are popular for unsupervised deep learning when the number of labelled examples is small, while deep convolutional neural networks (ConvNets) are popular for feature learning and supervised classification [17]. Such promising results motivate the application of deep learning techniques into the crack detection problems.

Successful application of deep learning techniques for crack detection rely on discriminative and representative deep features. In this paper, we develop a novel crack detection method in which the discriminative features are learned directly from raw image patches using the ConvNets. To the best of our knowledge, this work is the first attempt to bridge the gap between deep convolution neural networks and transportation research. The proposed approach differs from recent works on crack detection in the following four important aspects: 1) The proposed approach leverages deep learning based detectors instead of filter-based detectors as in [3]; 2) It does not make any assumption of the geometry of the pavement as required in [10]; 3) We use discriminative features, which are automatically learned from images, rather than hand-crafted features [8, 10]; 4) Unlike existing methods that require specific optical devices [5, 9], the proposed approach is successfully applied to images that are collected using a low-cost smart phone with complex background.

2. PROPOSED METHOD

Given a pavement image, the objective of a crack detection problem is to determine whether a specific pixel is a part of a

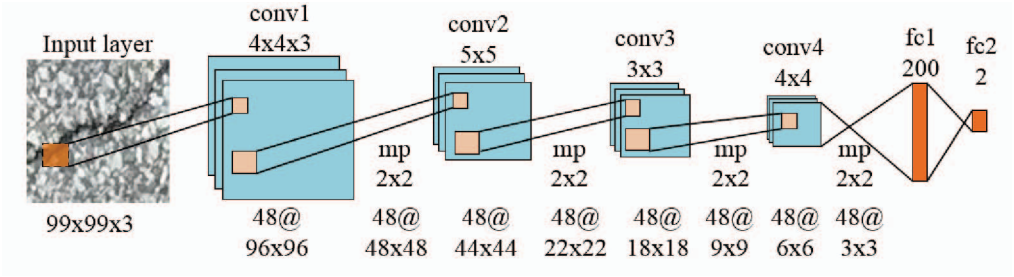

Fig. 1: Illustration of the architecture of the proposed ConvNet.

crack. To solve this problem, the proposed solution is based on a ConvNet, which is trained on square image patches with given ground truth information, for the classification of patches with and without cracks. For notational convenience, crack and non-crack patches are also referred to as positive and negative patches, respectively. In this paper, a patch whose center is itself a crack pixel, or is within the close vicinity of a crack pixel, is considered as a positive patch. Otherwise, this patch is considered as a negative patch.

2.1. Data preparation

Data set with more than 500 pavement pictures of size 3264 $\times$ 2448 are collected at the Temple University campus by using a smart phone as the data sensor. Each image is annotated by multiple annotators. In this study, to achieve a good compromise between computational cost and accuracy of the detection results [12, 13], each sample is a 3-channel (RGB) 99$\times$99 pixel image patch generated by the sampling strategy described in the following steps:

1. A patch whose center is within $f = 5$ pixels of the crack centroid is regards as a positive patch; otherwise it is considered as a negative patch.

2. To reduce the similarity between training samples, the overlap of two positive patches P_1 and P_2, expressed as $O = \text{area}(P_1 \cap P_2)/\text{area}(P_1 \cup P_2)$, should be kept at a low level. In this study, we choose the distance between the centers of two adjacent patches to be d=0.75w, where w is the width of a patch. For the negative patches, two adjacent patches should have no overlap.

3. Given a patch center c, each candidate patch is rotated around c by a random angle $\alpha \in [0°, 360°]$. This plays an important role to increase the number of crack samples because crack patches only consist of a small proportion of the collected images.

Out of the generated samples from the above steps, 640,000 samples are used as the training set, 160,000 samples are used as the validation set for cross-validation when training the ConvNets, and 200,000 samples are used as the testing samples. The numbers of crack and non-crack patches are set to equal in all three data sets.

2.2. ConvNet Architecture

The architecture of the ConvNet is illustrated in Fig. 1, where **conv**, **mp**, and **fc** represent convolutional, max-pooling and fully-connected layers, respectively. In general, the ConvNet is considered as a hierarchical feature extractor, which extracts features of different abstract levels and maps raw pixel intensities of the crack patch into a feature vector by several fully connected layers. All parameters are jointly optimized through minimization of the misclassification error over the training set via the back propagation method [18].

All convolutional filter kernel elements are trained from the data in a supervised fashion by learning from the labeled set of examples introduced in Section 2.1. In each convolutional layer, the ConvNet performs max-pooling operations in order to summarize feature responses across neighboring pixels. Such operations allow the ConvNet to learn features that are spatially invariant, i.e., they do not change with respect to the location of objects in the images. Finally, fully-connected layers are used for classification. Due to the mutually exclusive property of the underlying crack detection problem (crack or non-crack), a softmax layer is used as the last layer of the ConvNets to compute the probability of each class given an input patch.

Given a training set $S = \{x^{(i)}, y^{(i)}\}$ which contains m image patches, where $x^{(i)}$ is the i-th image patch and $y^{(i)} \in \{0, 1\}$ is the corresponding class label. If $y^{(i)} = 1$, then $x^{(i)}$ is a positive patch, otherwise $x^{(i)}$ is a negative patch. Let $z_j^{(i)}$ be the output of unit j in the last layer for $x^{(i)}$. Then, the probability that the label $y^{(i)}$ of $x^{(i)}$ is j can be calculated by

$$p(y^{(i)} = j | z_j^{(i)}) = \frac{e^{z_j^{(i)}}}{\sum_{l=1}^{k} e^{z_l^{(i)}}}, \tag{1}$$

and the corresponding cost function is given by

$$J = -\frac{1}{m} \left[\sum_{i=1}^{m} \sum_{j=1}^{k} 1 \left\{ y^{(i)} = j \right\} \log \frac{e^{z_j^{(i)}}}{\sum_{l=1}^{k} e^{z_l^{(i)}}} \right] \tag{2}$$

where $k = 2$, m is the total number of the patches, and $1\{\cdot\}$ stands for the indicator function.

2.3. ConvNet Training

The goal of training a ConvNet is to increase the variation of the training data and to avoid overfitting analogous to the training data set. The dropout method is used between two fully connected layers to reduce overfitting by preventing complex co-adaptations on training data [19]. The output of each neuron is set to zero with a probability of 0.5.

The training of the ConvNet is accelerated by graphics processing units (GPUs). Further speed-ups are achieved by using rectified linear units (ReLU) as the activation function [14], which is more effective than the hyperbolic tangent functions $\tanh(x)$ and the sigmoid function $(1+e^{-x})^{-1}$ used in traditional neuron models, in both training and evaluation phases. The ConvNets are trained using the stochastic gradient descent (SGD) method with a batch size of 48 examples, momentum of 0.9, and weight decay of 0.0005. Less than 20 epochs are needed to reach a minimum on validation set.

2.4. Processing a Testing Image

To process a testing image, the ConvNet can provide each point centered within the image a probability of being a crack or non-crack. This procedure yields a probability map. Inspired by the method proposed in [11], the probability of a point can be calculated by averaging probability $\{P_1, ..., P_N\}$ of each patch generated by randomly rotating it around its center pixel c, i.e.,

$$p(c|\{P_1(c), ..., P_N(c)\}) = \frac{1}{N}\sum_{i=1}^{N} P_i(c), \tag{3}$$

where $P_i(c)$ is the classification probability of the ConvNet computed for the i-th individual patch, and N is set to 5 for a computing efficiency. The ConvNet has a higher number degrees of freedom and thus tends to exhibit a large variance and a small bias [13]. As such, the number of crack patches are far less than that of background patches in an image. This fact makes the ConvNet to be likely to overestimate the crack probability. Therefore, an appropriate threshold has to be used. Define the precision and recall as

$$P = \frac{\text{true positive}}{\text{true positive} + \text{false positive}}, \tag{4}$$

$$R = \frac{\text{true positive}}{\text{true positive} + \text{false negative}}. \tag{5}$$

Then, the F_1 score is expressed as

$$F_1 = \frac{2PR}{P+R}. \tag{6}$$

The threshold used to re-estimate the final probability is determined such that it yields the largest F_1 score on the validation data set [13]. In this study, the threshold t is set to 0.64, at which the F_1 score is maximized.

Table 1: Hand-crafted features of image patches

Feature Descriptions	Number
Mean RGB	3
HSV for mean RGB	3
Hue histogram	5
Saturation histogram	3
LBP	59
Texton histogram	20

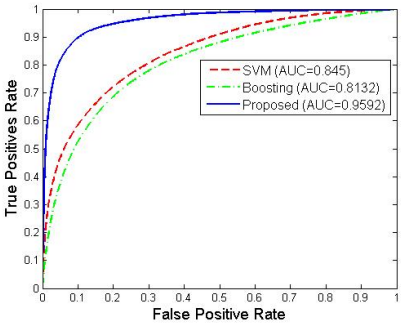

Fig. 2: ROC curves.

3. EXPERIMENTAL EVALUATION

All experiments are performed using an Intel(R) Xeon(R) E3-1241 V3 @ 3.5GHz CPU with 8 GB RAM and NVidia Quadro K220 GPU. The ConvNet was constructed via the Caffe [20] framework and trained by using 5-fold cross-validation. The proposed method is compared against the support vector machine (SVM) and the Boosting methods. The SVM is trained with LIBSVM [21] and the Gaussian radial basis function (RBF) kernel is used with C and γ determined using 5-fold cross-validation. The Boosting method [22] composed of 100 weak classifiers with a maximum depth of 5 is trained via the OpenCV toolkit. All parameters with the minimal test error of 5-fold cross-validation is used for comparison. The features for training the SVM and the Boosting are based on color and texture of each patch which are associated with a binary label indicating the presence or absence of cracked pavement. The feature vector is 93-dimensional, and is composed of color elements, histograms of textons and LBP descriptor within the patch. The detailed description of the feature vector is shown in Table 1. Some of the features are adopted from [23] and [10]. Different from [10], the geometry information is not considered in this work, since we aim to provide a crack detection method without specific geometry information. The Receiver operating characteristic (ROC) curves are shown in Fig. 2 and a summary of the statistics is given in Table 2. It is clear from these results that the ConvNet outperforms the other two detectors.

Figs. 3 and 4 show the images, together with the respective probability of correct classification, of selected patch-

Table 2: Performance comparison of different methods

Method	Precision	Recall	F_1 score
SVM	0.8112	0.6734	0.7359
Boosting	0.7360	0.7587	0.7472
ConvNets	**0.8696**	**0.9251**	**0.8965**

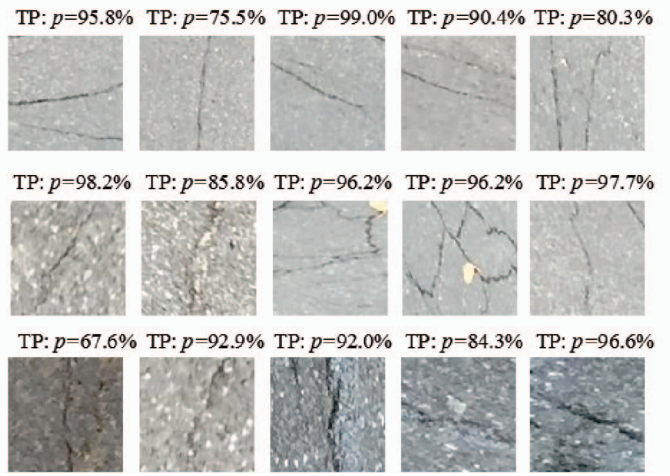

Fig. 3: Detection of crack: test probabilities of the ConvNet for being crack. TP denotes true positive.

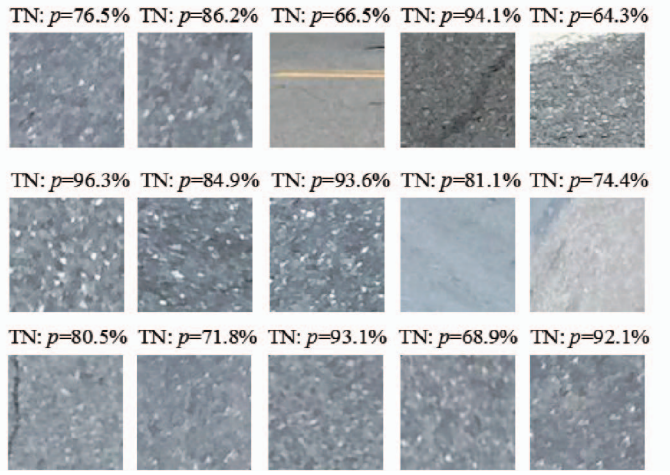

Fig. 4: Detection of non-crack: test probabilities of the ConvNet for being non-crack. TN denotes true negative.

es that are only correctly classified by the proposed method based on ConvNet. These results evidently demonstrate that the discriminative features learned from the ConvNet outperform the hand-crafted features in describing complex patch context.

We further compare the proposed method with the SVM and the Boosting methods using images of size 300×300. Cracks are detected by the trained ConvNet, SVM and Boosting method on a sliding window with step of 1 pixel. If a

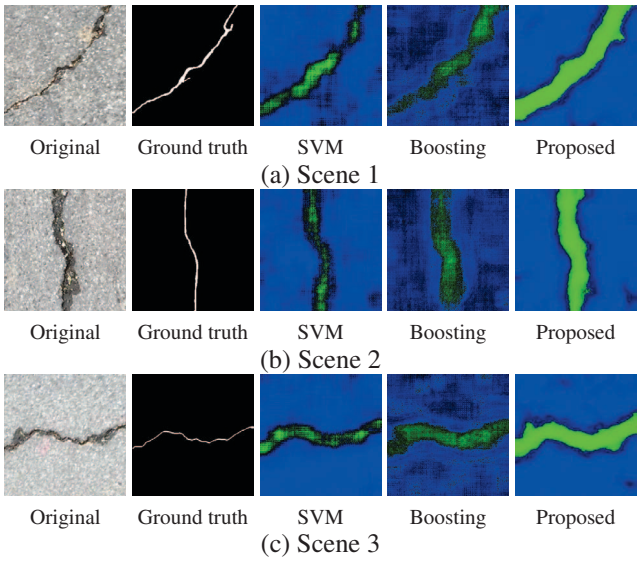

Fig. 5: Probability maps.

window lies partly outside of the image boundary, the missing pixels are synthesized by mirroring. Fig. 5 shows the crack detection results for three different scenes. For each scene, each row shows the original image with crack, ground truth, probability maps generated by the SVM and the Boosting methods, and that by the ConvNet. The pixels in green and in blue denote the crack and the non-crack, respectively, and a higher brightness means a higher confidence. The SVM cannot distinguish the crack from the background, and some of the cracks have be misclassified. Compared to the SVM, the Boosting method can detect the cracks with a higher accuracy. However, some of the background patches are classified as cracks, resulting in isolated green parts in Fig. 5. In contrast to these two methods, the proposed method provides superior performance in correctly classify crack patches from background ones.

4. CONCLUSIONS

We proposed an automatic detection method based on deep convolutional neural networks in which the features are automatically learned from manually annotated image patches acquired by a low-cost sensor, i.e., smart phone. To the best of our knowledge, this is the first study that applies deep-learning based method to road crack detection problem. In the future, we will optimize the proposed detection method and build an integrated low-cost system for real-time road crack detection.

5. ACKNOWLEDGEMENTS

The first author would like to thank Dr. Wangmeng Zuo and Dr. Feng Li from Harbin Institute of Technology for their helpful discussions.

6. REFERENCES

[1] A. Jahangiri, H.A. Rakha, and T.A. Dingus, "Adopting machine learning methods to predict red-light running violations," in *Proceedings of IEEE International Conference on Intelligent Transportation Systems*, Sept. 2015, pp. 650–655.

[2] A. Jahangiri and H.A. Rakha, "Applying machine learning techniques to transportation mode recognition using mobile phone sensor data," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 5, pp. 2406–2417, 2015.

[3] M. Salman, S. Mathavan, K. Kamal, and M. Rahman, "Pavement crack detection using the gabor filter," in *Proceedings of IEEE International Conference on Intelligent Transportation Systems*, Oct. 2013, pp. 2039–2044.

[4] 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, 2012.

[5] H. Oliveira and P.L. Correia, "Crackit-an image processing toolbox for crack detection and characterization," in *Proceedings of IEEE International Conference on Image Processing*, Oct. 2014, pp. 798–802.

[6] S. Mathavan, K. Kamal, and M. Rahman, "A review of three-dimensional imaging technologies for pavement distress detection and measurements," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 5, pp. 2353–2362, 2015.

[7] R. Medina, J. Llamas, E. Zalama, and J. Gomez-Garcia-Bermejo, "Enhanced automatic detection of road surface cracks by combining 2d/3d image processing techniques," in *Proceedings of IEEE International Conference on Image Processing*, 2014, pp. 778–782.

[8] Y. Hu and C. Zhao, "A local binary pattern based methods for pavement crack detection," *Journal of Pattern Recognition Research*, vol. 5, no. 1, pp. 140–147, 2010.

[9] 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, 2013.

[10] S. Varadharajan, S. Jose, K. Sharma, L. Wander, and C. Mertz, "Vision for road inspection," in *Proceedings of 2014 IEEE Winter Conference on Applications of Computer Vision*, 2014, pp. 115–122.

[11] H. Roth, L. Lu, J. Liu, J. Yao, A. Seff, C. Kevin, L. Kim, and R. Summers, "Improving computer-aided detection using convolutional neural networks and random view aggregation," *IEEE Transactions on Medical Imaging*, 2015.

[12] D. Ciresan, A. Giusti, L. M Gambardella, and J. Schmidhuber, "Deep neural networks segment neuronal membranes in electron microscopy images," in *Advances in Neural Nnformation Processing Systems*, 2012, pp. 2843–2851.

[13] D. C Cireşan, A. Giusti, L. M. Gambardella, and J. Schmidhuber, "Mitosis detection in breast cancer histology images with deep neural networks," in *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pp. 411–418. 2013.

[14] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Nnformation Processing Systems*, 2012, pp. 1097–1105.

[15] Y. Zhang, K. Sohn, R. Villegas, G. Pan, and H. Lee, "Improving object detection with deep convolutional networks via Bayesian optimization and structured prediction," in *Proceedsing of IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 249–258.

[16] J.J. Kivinen, C. K. Williams, and N. Heess, "Visual boundary prediction: A deep neural prediction network and quality dissection," in *Proceedings of International Conference on Artificial Intelligence and Statistics*, 2014, pp. 512–521.

[17] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[18] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[19] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[20] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," *arXiv preprint arXiv:1408.5093*, 2014.

[21] C. Chang and C. Lin, "Libsvm: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 3, pp. 27, 2011.

[22] Y. Freund and R. Schapire, "A short introduction to boosting," *Journal-Japanese Society For Artificial Intelligence*, vol. 14, no. 771-780, pp. 1612, 1999.

[23] D. Hoiem, A. Efros, and M Hebert, "Geometric context from a single image," in *Proceedings of International Conference on Computer Vision*, 2005, vol. 1, pp. 654–661.