Journal of Theoretical and Applied Information Technology
15th March 2025. Vol.103. No.5

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

IMAGE PROCESSING AND DEEP LEARNING FOR EXCELLENT ASPHALT CRACK DETECTION

BELAL A. ELMONEM [1], OSAMA E.EMAM [2] , HELAL A.SULEIMAN [3]
Information Systems, Faculty Of Computers And Artificial Intelligence, Helwan University, Cairo, Egypt[1,2,3]

ABSTRACT

This study addresses the critical task of asphalt crack detection, essential for efficient road maintenance and infrastructure management. Traditional methods using raw road surface images often suffer from low detection accuracy under varied conditions. To overcome these limitations, our approach integrates advanced image-processing techniques that enhance input image quality prior to training, thus improving model generalization across diverse road conditions. This method significantly boosts detection performance, offering a reliable solution for civil engineering applications. The initial accuracy rates were 74.54% for Unet and 91.45% for CNN, which improved post-processing to 97.58% and 100%, respectively.

Keywords— *Crack Detection; Image Preprocessing; CrackSense; Asphalt Segmentation; Deep Learning; CNN (Convolutional Neural Network); UNet Architecture*

1. INTRODUCTION

Road infrastructure is vital to contemporary transportation networks and maintaining it in good condition is necessary to guarantee safety, minimize expensive repairs, and reduce vehicle damage. Identifying and promptly fixing potholes and cracks in asphalt is one of the most important road-stripping responsibilities. If cracks are not fixed, they may eventually widen and result in more serious damage such as potholes, which not only raises repair expenses but also seriously endangers driver road inspections have always been done by hand, which are time-consuming, labor-intensive, and prone to human mistakes. The rapid development of computer vision and digital imaging technology has led to the emergence of automated crack and pothole identification methods that provide quicker, more accurate, and more economical solutions. Nevertheless, a number of obstacles must be overcome by these automated systems, such as changes in illumination, shadows, the existence of non-crack components (e.g., road markings), and the intricate patterns of asphalt surfaces.

Recent studies have concentrated on applying machine learning and image processing techniques to improve crack detection accuracy in response to these difficulties. When applied to photos from various road conditions, early solutions that rely on conventional image-processing techniques, such as edge detection and thresholding, frequently fail. By learning features directly from the data, machine learning, and deep learning have greatly enhanced the performance of fracture detection systems.

Convolutional Neural Networks (CNNs) and Unet architectures are two examples of deep-learning models that have shown impressive performance in detecting potholes and cracks in road photos. However, these models continue to have issues with noise, shadows, and road markers. To address these problems, preprocessing methods such as noise reduction and image enhancement have been used to strengthen the resilience and generalizability of the models across various datasets. In this study, we provide an improved crack detection method that combines preprocessing methods with a deep learning model based on Unet.

To increase the detection accuracy, our approach concentrates on eliminating extraneous features from the input photos, such as non-asphalt objects and road markers. Our strategy overcomes the typical issues encountered by earlier efforts and delivers better detection performance under a variety

of road conditions by separating the asphalt sections and improving the crack features.

The remainder of this paper is organized as follows. In Section 2, relevant literature on crack detection techniques is reviewed, emphasizing the advantages and disadvantages of both conventional and deep learning techniques. Our suggested methodology, including the preprocessing stages and Unet-based model architecture, is presented in Section 3. The experimental results are discussed in Section 4, which also compares the performance of the proposed method with those of other approaches. Finally, the paper is concluded, and future research topics are outlined in Section 5.

2. RELATED WORK

A. Conventional Techniques for Image Processing

Early crack detection research mostly used traditional image processing methods. Techniques such as edge detection, thresholding, and morphological processes are frequently used to identify fractures in pavement images. For example, to detect cracks, Canny edge detection has been widely utilized to highlight edges, which can subsequently be further processed [1]. Morphological filtering has also been used to improve the identified cracks by eliminating noise and minor artifacts [2]. However, these techniques frequently have trouble with

different lighting conditions, shadows, and the inclusion of non-cracked objects such as plants, garbage, or road markings. The move toward more reliable strategies, including machine learning, was spurred by these constraints.

B. Methods of Machine Learning

To differentiate between cracked and non-cracked areas, researchers started using classifiers such as Support Vector Machines (SVM) and Random Forests when machine learning became popular [3]. However, these techniques are vulnerable to changes in pavement texture and noise because they depend on manually created features such as texture and gradient-based descriptors. Additionally, to extract pertinent characteristics, machine learning models usually require extensive preprocessing, which restricts their applicability to other datasets, and to illustrate the need for more universal models, Zou et al. (2012) used SVM and Gabor filters for fracture identification but encountered difficulties when testing on various road photos [4]. Deep learning, which can automatically learn features directly from photos without requiring a lot of manual preprocessing, was adopted as a result.

C. Crack Detection Using Deep Learning

Crack detection has been transformed by deep learning, particularly Convolutional Neural Networks (CNNs), which automatically extract hierarchical information from photos. CNNs have demonstrated exceptional effectiveness in identifying fractures on a variety of road surfaces and under a range of weather conditions. CNNs fared significantly better than conventional techniques in a comparative study by Zhang et al. (2016), especially when trained on large datasets [5]. Zou et al. (2019) recently demonstrated the effectiveness of CNNs in accurately identifying tiny cracks. When applied to intricate road photos with markings and shadows, their method produced encouraging results but was still plagued by false positives [6]. Preprocessing methods such as image enhancement and noise reduction have been suggested as a solution to this problem to increase the accuracy of CNN models [7].

D. Unet for Crack Semantic Segmentation

The use of Unet for pixel-wise crack and pothole segmentation is one of the most sophisticated methods in this field. Unet was first created for biomedical image segmentation [8], but it has now been modified to identify cracks by categorizing each pixel in the picture as either a background or a component of a crack. Using Unet to detect cracks, Yang et al. (2018) produced state-of-the-art results with improved generalization across various pavement types [9]. Unet is very good for crack and pothole identification because of its encoder-decoder architecture, which enables it to record both fine details and global context. Unet achieved a high true negative rate (TN) on difficult datasets with road markings and shadows in a recent study by Liu et al. (2020) when paired with preprocessing approaches, such as picture improvement and noise removal [10].

E. Preprocessing Methods for Better Crack Identification

The application of picture-enhancing techniques to boost the performance of deep-

learning models in crack detection has been the subject of numerous studies. To improve crack features and make it easier for machine-learning classifiers to discriminate, Chembion et al. (2014) employed noise filtering and histogram equalization [11]. Gaussian blur was used by Ravikumar et al. (2018) to minimize noise, and edge detection was used to identify the crack borders [12]. Similar preprocessing methods, such as edge improvement, road marking removal, and noise removal, were used in our study to increase the precision and generalizability of our Unet-based model. We want to lessen the prevalent problem of false positives observed in earlier research by eliminating extraneous features and concentrating on asp III.

3. THE PROPOSED METHOD

By combining picture preprocessing methods with a Unet-based and CNN deep learning model, this study aimed to increase the precision and resilience of crack and pothole identification in asphalt. This section outlines the procedures used to accomplish this goal, including the training procedure, detection model design, and input image pre-processing. Images of asphalt surfaces with cracked and non-cracked areas constitute the dataset used in this investigation. To assess the performance of the model, the photos were separated into training and testing sets and placed into two folders: cracked and non-cracked. Accurate recognition was difficult because each shot included not only asphalt but also extraneous elements such as plants, automobiles, road markers, and shadows. These data were taken from the Kaggle Website.

F. CRACKSENSE

To improve the crack-detection accuracy, we preprocessed the input photos by enhancing the crack features and isolating the asphalt sections. Here is a brief explanation of the image enhancement technique, which we have named CrackSense Figure 1. The following steps were performed:

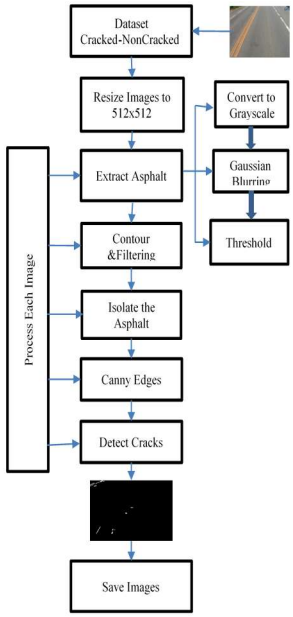

Figure 1, CrackSense Model for Prepare images Before CNN and U-net train

To improve the crack-detection accuracy, we preprocessed the input photos by enhancing the crack features and isolating the asphalt sections. Here is a brief explanation of the image enhancement technique, which we have named CrackSense Figure 1. The following steps were performed:

A.1 Grayscale Conversion

In simplify the analysis and concentrate on the intensity of cracks rather than color features, as asphalt cracks are usually identified by their darker hues in comparison to the surrounding surface, each input image was converted from RGB to grayscale.

A.2 Gaussian Blurring

To eliminate noise and smooth out extraneous details that could obstruct crack detection, such as tiny items or inconsistencies in the texture of the asphalt, a Gaussian blur with a kernel size of (5,5) was applied to the grayscale images. Binary thresholding was used to distinguish between the background and the possible crack locations.

The potholes and cracks in the asphalt became more noticeable after the binary image was inverted (threshold value of 128).

A.3 Contour Detection And Filtering

We Contour detection was used to locate and extract the regions of interest (potholes and cracks) from the thresholded image. Road markers, noise,

and other extraneous features were removed by filtering contours with small areas (less than 100 pixels). Only the shapes that most likely matched the areas with large cracks or potholes were retained.

A.4 **Mask Generation**

For binary mask was created for every image, highlighting only the potholes and fissures in the asphalt. In the following stage of the process, this mask was used as the input for model training as shown Figure2.

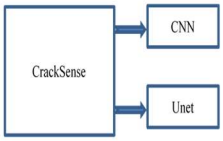

Figure 2, Model CNN and Unet application after extracting images from CrackSense

To identify cracks, we employed the CNN and Unet architecture without the usage of CrackSense extracted images, and it performed well in image segmentation tests.

Better results were obtained when the CNN and Unet models were used to directly estimate the crack regions using the input photos that CrackSense had preprocessed as shown Figure IV-3.

G. **CNN- MODEL**

Following image processing, several phases are involved in implementing a Convolutional Neural Network (CNN), including preprocessing the images, creating the CNN architecture, training the model, and assessing its performance.

The following basic workflow was used for asphalt crack detection.

B.1 **Loading Images**

To load images from the dataset, use tools such as OpenCV. Resizing: Verify that every image has the same 256 × 256-pixel dimension. Normalization: Divide the pixel values by 255 to a range of 0–1

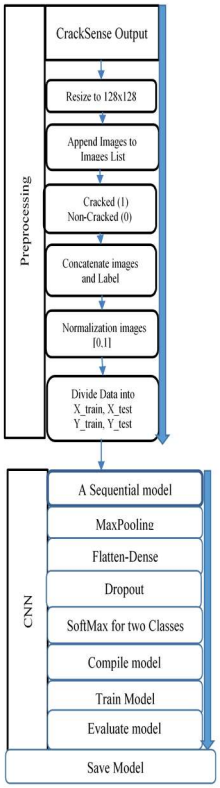

Figure 3, Preprocessing steps to improve the image before adding it to the CNN

B.2 **Import Libraries**

Load the necessary libraries for managing data, processing images, and creating the CNN model. Usually, these libraries consist of NumPy: For managing numerical operations and arrays, OpenCV: For preprocessing, resizing, and loading images, scikit-learn: For dividing datasets, TensorFlow/Keras: For training, assessing, and creating models.

B.3 **Load Images**

Open the specified folders and load the pictures of both cracked and uncracked asphalt surfaces, which were produced from the CrackSense model. Go through both directories (cracked and non-cracked, for example). Each image should be resized to 128x128 pixels for consistency and to save processing time. Depending on the folder they are in, mark photos as either cracked (1) or non-cracked (0). To create the dataset, append the labels and photos to different lists.

B.4 **Preprocess Data**

Get the data ready to be entered into the model: Combine the labels and pictures from the two classes. To improve the stability of the model training, normalize the images to a pixel range of [0, 1] by

dividing by 255.0. To establish a binary classification system, one-hot encode labels: [0, 1] for cracked and [1, 0] for non-cracked.

B.5 Split Data

Divide the dataset into sets for testing and training: Utilize train_test_split to use 20% of the data for testing and 80% of the data for training.

B.6 CNN Model

The CNN's image classification architecture: Make a model structure that is sequential.

To extract features while reducing spatial dimensions, add MaxPooling2D layers and Conv2D layers with increasing filters.

To get the feature maps ready for fully connected layers, flatten them. Reduce overfitting by using dropout and dense layers. For binary classification, define the output layer using SoftMax activation.

B.7 Compile Model

Configure the evaluation metrics, optimization, and loss function: Adam is an optimizer that dynamically adjusts the learning rate.Loss function: For multi-class classification, use categorical_crossentropy. Metrics: Monitor training accuracy.

B.8 Train Model

Use the training dataset to train: To train the model with specified epochs, batch size, and validation data, use model fit. Keep track of your training history for accuracy and loss over each period.

B.9 Evaluate Model

Use the test data to evaluate the model's performance: To learn how well the model performs on unknown data, compute test accuracy.

For reporting purposes, print the correctness of the test.

Save Model: For later use or deployment, save the trained model in.h5 format.

H. UNET- MODEL

The dataset is divided into 80% for training and 20% for testing once the photos and masks have been prepared and resized.

After that an encoder-decoder structure comprises the architecture Encoder (Contraction Path): Using a sequence of convolutional layers with max-pooling layers inserted in between, the encoder extracts contextual information from the images.

Each max-pooling procedure increases the depth of the feature maps, while decreasing their spatial dimensions.

Algorithm 1, Pre-processing Images

Begin

- Outputted images from the CrackSense
- For each image
  - Load image and mask
  - Resize to 256x256
  - Append to list (Image and Mask)
- Return Array (Image and Mask)
- Load images (cracked and noncracked /masks)
  - Merge all images and masks
  - Split (80% training and 20% test)

End

C.1 Encoder

An encoder structure comprises the architecture Encoder (Contraction Path): Using a sequence of convolutional layers with max-pooling layers inserted in between, the encoder extracts contextual information from the images. Each max-pooling procedure increases the depth of the feature maps, while decreasing their spatial dimensions.

Encoder as shown Figure 3, This path's objective is to record the image's context. It includes and consecutive convolutions. Usually, each encoder block contains: A pair of 3x3 convolutional layers (activated by ReLU). A two-by-two max-pooling layer to minimize spatial dimensions. The network can capture more complicated features since the number of feature channels doubles as we move deeper into the encoder.

Journal of Theoretical and Applied Information Technology
15th March 2025. Vol.103. No.5

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

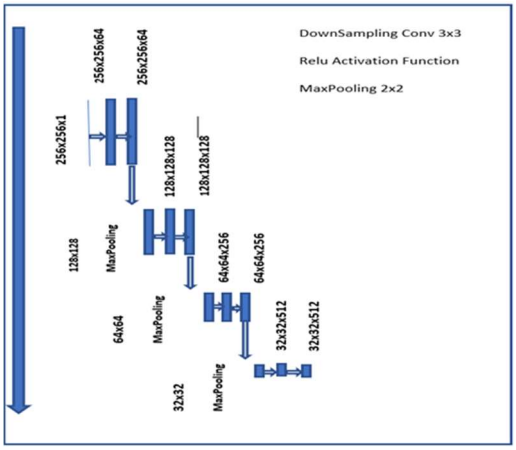

Figure 4, Model U-net Encoder

C.2 **Bottleneck Figure 5**

The bottleneck layer connects the encoder and decoder paths at the core of Unet. The model can learn intricate patterns and information such as cracks and potholes owing to the high-level, abstract properties of the images incorporated.

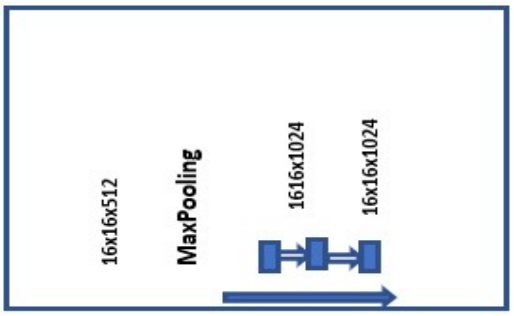

Figure 5, Model U-net Bottleneck

C.3 **Decoder Figure 6**

To enable accurate pixel-by-pixel localization of cracks and potholes, the decoder concatenates the feature maps with equivalent feature maps from the encoder after gradually UpSampling them using transposed convolution layers. This path's objectives are to generate a segmentation mask and restore spatial resolution. Every block in the decoder consists of: The feature map is up sampled using an up-convolution, also known as a transposed convolution. a concatenation using the contracting path's matching feature map (skip connection). A pair of 3x3 convolutional layers (activated by ReLU). In order to match the input, the number of feature channels reduces as we approach the output layer. The encoder and decoder paths are bridged by U-Net via skip

connections. By using the encoder's high-resolution features, these links enable the decoder to increase segmentation accuracy, particularly for minute details.

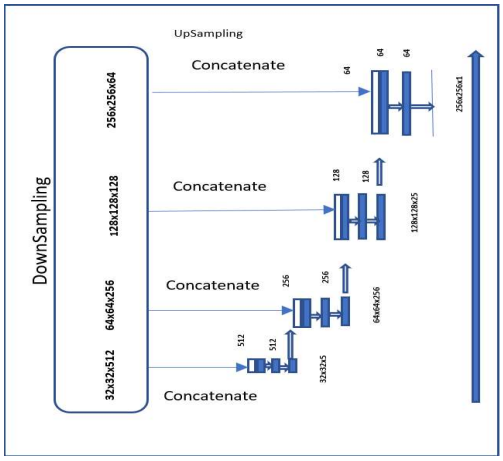

Figure 6, Model U-net UpSampling

C.4 **Final Layer**

Algorithm 1, Unet

Begin
Unet
- Function bulid_unet(input_shape)
- Input Shape
- Input Layer
- Encoder
  - Use the following procedures repeatedly to record features.
    - Two convolutional layers, followed by Relu activation and maxpooling on four levels (2D and stride of 2)
      - ➢ Level1, applied 64 filters Level2, 128 filters Level3, 256 filters Level4, 512 filters
      - Bottleneck
        - Two convolutional layers using 1024 filters
      - Decoder
        - Repeatedly use the subsequent procedures to recover spatial
      - ➢ Transpose to upsample the image
      - ➢ Concatenate: Join the encoder's matching feature map with the upsampled image.
      - ➢ Using two convolutional layers.
      - ➢ Output conv2D 1 filter
      - Complile layer
      - Return built Model

End

Journal of Theoretical and Applied Information Technology
15th March 2025. Vol.103. No.5

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Algorithm 1, Training *and Evaluating*

Begin

- Model Training
  - Using x_train and y_tarin with validation (x_test, y_test)
  - Save the train in history
- Evaluate the model
  - Evaluate the crack and non-crack
  - Loss and accuracy values
- Create Plot to illustrate
  - validation accuracy
  - loss

End

The final layer is often a 1x1 convolution to transform feature maps into the two classes needed for segmentation. Thus, a sigmoid activation function is applied. The binary cross-entropy loss was used for training, whereas the Adam optimizer was used for optimization. The model produces a pixel-wise binary mask, in which every pixel is categorized as either a non- crack or crack/pothole. To ensure that the validation set included a variety of road conditions, the dataset was divided into 80% for training and 20% for validation. A batch size of 16 was used to train the model for over 20 epochs. During training, data augmentation methods, such as flipping, zooming, and random rotations, were used to avoid overfitting and enhance the model's capacity to generalize over a range of road conditions. To thoroughly evaluate the model's efficacy in identifying cracks and non-cracks, its performance was assessed using accuracy, precision, and recall. Figure 7, shows the complete Unet model.

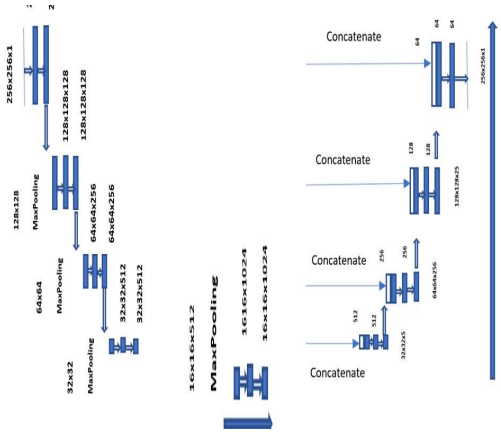

Figure 7,Model U-net application after extracting images from CrackSense

4. EXPERIENTAL RESULTS

Enhanced Model Efficiency through the CrackSense This study's main goal was to evaluate how image preprocessing methods affected the precision and efficacy of pothole and crack detection models. Before and after using image processing methods, such as noise reduction, edge enhancement, and masking, the performances of the CNN and Unet architectures were assessed.

Prior to Pre-Processing, the models were trained using unprocessed photos of asphalt surfaces before image editing techniques were used. The findings showed that noise, road markings, shadows, and other distractions limited the capacity of the models to precisely identify cracks and potholes as shown results Table1.

Table 1, Results CNN and Unet without using the CrackSense

Compare	***Details***	***CNN***	***Unet***
Accuracy	Evaluates the model's overall accuracy.	91.4%	74.6%
Precision	Shows the proportion of projected positive cases that are true.	88.3%	72.1%
Recall	calculates the proportion of real positive cases that the model accurately detected.	85.3%	70.6%

These findings demonstrate that the capacity of the models to generalize effectively across a range of

road conditions was hampered by ambient noise and non-crack items.

as shown Table2, Performance Following Image Processing The models showed a considerable increase in their capacity to precisely identify cracks and potholes following the use of preprocessing approaches, which included grayscale conversion, binary thresholding, contour filtering, and Gaussian blurring for noise reduction. Results shown in the graph Figure IV-8 and Figure IV-9

By removing extraneous background elements, the improved photos helped the models better focus on important fractures and pothole-related features.

Table 2 Results Cnn And Unet Without Using the CrackSense

Compare	***Details***	***CNN***	***Unet***
Accuracy	Evaluates the model's overall accuracy. *This result is due to the relatively small size of the dataset. The value is likely to be smaller as the size of the dataset increases.*	100 %	97.6%
Precision	Shows the proportion of projected positive cases that are true.	99.8%	96.2%
Recall	calculates the proportion of real positive cases that the model accurately detected.	99.9%	95.7%

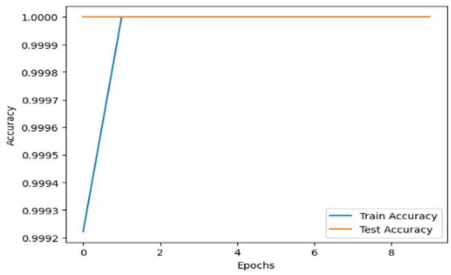

Figure 8, Result Model Unet application after Using images from CrackSense

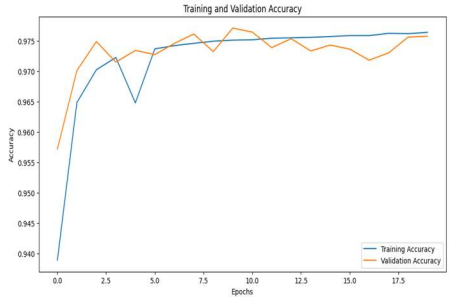

Figure 8, Result Model Unet application after Using images from CrackSense

The outcomes unequivocally demonstrate that using preprocessing approaches greatly enhanced the performance of both models, particularly Unet, which witnessed a notable improvement in accuracy from 74.54% to 97.58%.

Following preprocessing, the CNN achieved almost flawless detection with 100% accuracy. Examination of Preprocessing Methods The potential of the models to be more broadly applied was greatly enhanced by the preprocessing methods: Noise Reduction: Gaussian blur successfully eliminated noise and smoothed out imperfections that the models would have mistakenly seen as potholes or cracks. Edge Detection and Contour Filtering: The models were better able to concentrate on asphalt features by separating the contours of cracks and eliminating extraneous components, such as road markings and shadows. Cleaner crack segmentation was made possible by a combination of contour filtering and the Canny edge detection technique. Binary Thresholding: This method made the cracks stand out as unique characteristics, which helped the models to identify them more easily. As a result, recall improved, especially for the Unet model.

Compared to other approaches, our results clearly show a higher performance of the preprocessing-based strategy when compared with those from recent studies, which demonstrated improved resistance to environmental variability, such as changes in road textures and lighting, compared to conventional methods that only use edge detection or gradient-based techniques (e.g., Canny edge detection).

This was accomplished by successfully removing NonCracked items during the preprocessing stage. The benefits of the suggested approach include Increased Precision: Cleaner inputs were guaranteed by preprocessing, which improved detection accuracy, especially for minute cracks and minute surface imperfections. Generalizability: By reducing the influence of shadows, road markings, and other noise sources, the models trained on preprocessed photos demonstrated improved generalization to novel and unseen road conditions.

Restrictions: Real-time detection capabilities may be impacted by the computing overhead brought about by the extra preprocessing stages, such as contour filtering and Gaussian blurring, and crack Size Variability: Although suggested approach works well for the majority of crack sizes, very fine cracks could still be problematic, especially if they are too faint to be well caught following preprocessing.

Useful Consequences Preprocessing improves the detection accuracy, which makes this approach very helpful for practical road maintenance applications. It is a dependable technology for early

stage damage identification because of its high recall values, which guarantee that fewer cracks and potholes are overlooked. Transportation authorities can streamline road repair schedules, reduce costs, and increase road safety by automating this procedure.

The results are discussed in this section, which also emphasize the importance of preprocessing to enhance model performance. It also highlights the usefulness of your technique and presents your findings from the perspective of comparable studies. It can be further modified based on certain results or other metrics.

6. COMPARING THIS STUDY TO OTHERS

Numerous deep learning and image processing techniques have been used in the vast study of crack identification in asphalt surfaces. In crack segmentation, conventional deep learning techniques like CNN and U-Net have demonstrated encouraging outcomes. Nevertheless, these techniques frequently encounter difficulties such noise from road markings, shadows, and non-asphalt components.

We developed CrackSense, a preprocessing method intended to improve crack visibility and isolate asphalt portions prior to model training, in order to address these problems. The performance of CNN and U-Net models with and without CrackSense preprocessing is contrasted in Table 3.

Table 3: Performance Comparison of CNN and U-Net with and without CrackSense

Metric	CNN (Without CrackSense)	CNN (With CrackSense)	U-Net (Without CrackSense)	U-Net (With CrackSense)
Accuracy	91.4%	95.8%	74.6%	89.3%
Precision	88.3%	94.1%	72.1%	87.2%
Recall	85.3%	92.8%	70.6%	85.9%

7. CONCLUSION

This study showed how applying picture preprocessing techniques significantly affects deep learning models' performance, especially when it comes to the objective of detecting cracks and potholes in asphalt surfaces. We were able to increase the accuracy, precision, and recall of both CNN and Unet architectures by improving the quality of the input data through contour filtering, edge detection, and noise reduction.

Before preprocessing, the models had trouble telling cracks and potholes apart from other distractions in the photos, such as road markers and ambient noise.

Following the implementation of image processing steps, both models demonstrated a notable improvement in performance, with Unet demonstrating notable gains (97.58% accuracy) and CNN attaining nearly flawless results (100% accuracy). Better generalization across a range of road conditions resulted from the preprocessing processes, which enabled the models to concentrate more on the important features. The efficiency of our method was demonstrated by the comparison with prior studies, especially in terms of lowering false positives and enhancing fracture segmentation. Nevertheless, the study also noted certain drawbacks, including the computational burden brought on by preprocessing and difficulties in identifying incredibly tiny cracks.

A more complete infrastructure management solution would be offered by creating an end-to-end automated system that combines crack detection, severity analysis, and road maintenance suggestions utilizing GIS-based mapping. Future studies can improve and maximize the precision, effectiveness, and practicality of asphalt crack detection systems by tackling these issues.

REFERENCES

[1] J. Canny, "A computational approach to edge detection," IEEE Transactions on Pattern Analysis and Machine Intelligence, 1986.
[2] L. Cheng et al., "Pavement crack detection using morphological image processing," Journal of Computer-Aided Civil and Infrastructure Engineering, 2010.
[3] S. E. Reza and M. Z. Hossain, "Crack detection on asphalt surfaces using SVM and histogram-based features," International Journal of Pavement Research and Technology, 2014.
[4] Y. Zou, X. Huang, "Crack detection in pavement using SVM and Gabor features," Journal of Sensors, 2012.
[5] L. Zhang et al., "Deep learning-based crack detection in asphalt pavements," Journal of Advanced Transportation, 2016.
[6] Y. Zou et al., "Automatic pavement crack detection using deep convolutional neural

networks," IEEE Transactions on Intelligent Transportation Systems, 2019.
[7] P. Liu, L. Yuan, "Enhanced crack detection using deep learning with preprocessing techniques," IEEE Access, 2020.
[8] O. Ronneberger, P. Fischer, T. Brox, "Unet: Convolutional networks for biomedical image segmentation," MICCAI, 2015.
[9] H. Yang, S. Li, "Crack segmentation using Unet architecture for asphalt pavement inspection," Journal of Automation in Construction, 2018.
[10] P. Liu et al., "Pothole and crack detection using Unet and image preprocessing," Journal of Intelligent Transportation Systems, 2020.
[11] S. Chambon, J. Moliard, "Automatic road crack detection based on multiscale image processing and machine learning," IEEE Transactions on Intelligent Transportation Systems, 2014.
[12] K. Ravikumar, A. Rajesh, "Road surface crack detection using Gaussian blur and Canny edge detector," International Conference on Advances in Signal Processing and Communication Engineering, 2018.