DEEP LEARNING PIPELINE FOR MODELING PAVEMENT CRACKS WITH AN IMBALANCED DATASET (MOPAC)

Andrew Scouten
Texas State University
601 University Drive, San Marcos, Texas 78666-4684
Email: yzb2@txstate.edu

Haitao Gong, Ph.D.
Texas State University
601 University Drive, San Marcos, Texas 78666-4684
Tel: 512-245-1826; Email: h_g153@txstate.edu

Jelena Tešić, Ph.D.
Texas State University
601 University Drive, San Marcos, Texas 78666-4684
Tel: 512-245-3409; Email: jtesic@txstate.edu

Feng Wang, Ph.D., Corresponding Author
Texas State University
601 University Drive, San Marcos, Texas 78666-4684
Tel: 512-245-5258; Email: f_w34@txstate.edu

Word Count: 4317 words + 7 table(s) $\times$ 250 = 6067 words

Submission Date: August 1, 2024

ABSTRACT

Automated pavement crack detection through automatic scanning and processing of pavement surface images is a critical task for efficient assessment of pavement conditions and informed decision-making for pavement maintenance activities. Recently, the adoption of Deep Learning has significantly improved pavement distress detection performance. However, current datasets and studies in this field predominantly focus on standard distress classes, such as longitudinal and transverse cracks, while rare classes or detailed categorizations are often neglected. This study introduces the MoPac algorithm: a novel approach that leverages deep neural networks (DNNs) through a cascade modeling technique combined with low-shot learning, with a particular focus on imbalanced distress classes, to enhance the accuracy and efficiency of pavement crack detection. Our proposed method utilizes a multi-stage DNN cascade to progressively refine crack detection, starting from a broad classification stage and narrowing down to precise localization and severity assessment. The integration of low-shot learning facilitates practical training on limited datasets by incorporating transfer learning and data augmentation strategies. This hybrid approach not only improves detection performance on diverse pavement conditions but also reduces the need for extensive annotated data. Empirical results demonstrate significant advancements in the accuracy and robustness of MoPac detection compared to state-of-the-art methods.

Keywords: Automated pavement crack detection, Deep learning, YOLO, Imbalanced Dataset

INTRODUCTION

A pavement management system is crucial for maintaining highway pavements in good shape at a minimum cost—exposures to weathering and traffic loading lead to cracks in the pavements. The detection of pavement cracks ensures on-time repair and prolongs the life of the road. For example, in Texas, there are close to 73 million miles of roads and 18.3 million licensed drivers. The Texas drivers drive an average of over 15.2 thousand miles a year, so they cover close to *230 billion miles a year*. That means, on average, a single patch of Texas road is driven on more than 3100 times a year. That is a lot of usage, and efficient pavement crack detection and repair is crucial to keep drivers and passengers safe. Currently, automated pavement crack detection systems are employed to identify and address pavement issues. These systems utilize various sensors and image-processing techniques to detect cracks and other forms of distress. However, achieving accurate detection remains challenging due to several factors. The primary challenge arises from the complexities of various distress types, different pavement types, and different pavement conditions related to age, traffic load, environment, and other factors. Pavement distress encompasses a wide range of issues, from minor hairline cracks to severe structural defects. Roads are constructed using different materials, such as asphalt, concrete, and composite materials, each exhibiting unique background characteristics. The existing condition can further complicate the attributes of both distress and the pavement itself. Sensor acquisition also complicates the detection process. The diverse types of sensors used, ranging from high-resolution cameras to less sophisticated devices, contribute to inconsistencies in image quality and data richness. These discrepancies impact the ability of automated systems to perform consistently across different settings and conditions. Additionally, the distribution of distress classes varies in a wide range, which necessitates sophisticated detection methods capable of accurately classifying and assessing a wide range of crack types.

There has been a plethora of work in the field in the past decade, as described in Section 3. However, the proposed approaches fail to scale to real-life applications due to the following roadblocks: (1) the lack of open-source datasets and a unified significant benchmark for evaluating detection technologies, and (2) the inadequacy of models in detecting the full range of pavement distress classes. Without widely available datasets that represent diverse and realistic scenarios, it is challenging for researchers to train and test their models under varied conditions. This lack of standardization also makes it difficult to compare different approaches and technologies effectively, slowing down progress in the field. Furthermore, many current models fail to detect rare distress classes or provide detailed categorization, which can result in missing critical defects or miss-classifying the severity of the cracks, leading to inadequate maintenance responses.

The images used in pavement crack detection are distinct from consumer images captured in everyday settings, and we cannot expect out-of-the-box Deep Neural Network (DNN) solutions to work unless it is a highly controlled experiment. We have found that state-of-the-art models often struggle with pavement image variability, leading to incomplete or inaccurate assessments (*1*). There is a pressing need for advanced automated solutions that can address the specific demands of pavement crack detection in the images captured under realistic conditions. Thus, we propose the novel MoPac algorithm, short for the **Mo**deling **Pa**vement **C**racks. MoPac provides a more robust, adaptable, and standardized system. MoPac is a cascaded deep learning algorithm that incorporates improvements for pavement imagery to detect and identify cracks. The algorithm was named after a nickname for the Loop 1 Expressway in Austin, Texas, MoPac. MoPac sets a skeleton baseline and introduces an innovative framework for practical deployment in road maintenance and safety assessment. MoPaC will enhance the efficiency of road maintenance processes,

help extend the lifespan of infrastructure, and ultimately ensure safer and more reliable roadways for all users.

RELATED WORK

In the past couple of years, pavement crack detection research has solely focused on adapting and improving state-of-the-art DNN systems to identify cracks accurately and efficiently. The work stems from the significant advances in the field of computer vision in the past decade in object detection using DNN. Several of those DNN networks were shown to be quite reasonable in detecting cracks: *YOLO* (You Only Look Once) revolutionized the field with its unified approach, predicting bounding boxes and class probabilities directly from full images in a single forward pass, enabling real-time processing while maintaining high accuracy, developed by Redmon et al. (*2*). Subsequent versions of *YOLO*, such as *YOLO*v2 (*3*), *YOLO*v3 (*4*),*YOLO*v4 (*5*), and beyond have incorporated advanced techniques to achieve better performances, including multi-scale prediction, improved network architectures, and data augmentation methods. Among the *YOLO* family, *YOLO*v5 is widely adopted for applications and serves in many studies as the base model for developing new approaches. The latest version was proposed by Wang et al. (*6*), which proposed the concept of Programmable Gradient Information (PGI) for more reliable training. The *Mask R-CNN* utilizes a Region Proposal Network (RPN) to generate potential object regions, which are then refined by a Fast *R-CNN* detector. This technique achieves remarkable precision in real-time object detection and predicts segmentation masks in parallel with bounding box recognition He et al. (*7*). *Cascade R-CNN* further enhances detection performance by employing a multi-stage object detection architecture that progressively improves the quality of detected objects through a series of detectors trained with increasing IoU thresholds introduced by Cai and Vasconcelos (*8*). The *U-Net*'s encoder-decoder architecture enables precise localization Ronneberger et al. (*9*), and the *DeepLab* employs convolution and pyramid pooling modules to capture multi-scale context Chen et al. (*10*).

Many studies have been conducted to evaluate the effectiveness of state-of-the-art object detection methods for crack detection. These studies focus on the evaluation of detection performance over a specific dataset, with little or no modification to the methods themselves. Majidifard et al. (*11*) applied both *YOLO* v2 and *Faster R-CNN* on a dataset developed with Google Street View images. According to the results, *YOLO* v2 outperformed *Faster R-CNN*, achieving an F1 score of 0.84 compared to Faster R-CNN's score of 0.65. Among the studies based on another dataset that includes pavement images collected by smartphones in three different countries (*12*), YOLO-based models have also been found to outperform other methods, such as Faster R-CNN and EfficientDet (*13*). By comparing the studies using datasets from only one country and all three countries (*13*)(*14*), it is noticed that the performance dropped significantly with the larger dataset, which includes all three countries. A study using high-resolution (1,800 × 1,200) three-dimensional (3D) asphalt and concrete pavement images reported that the Faster R-CNN and YOLO v3 delivered the same level of performance for the automated distress detection and classification task, with both methods achieving around 90% accuracy (*15*). A study applied *YOLO* v5l to a two-dimensional (2D) asphalt concrete pavement set of 3,001 images (*16*), and they reported similar average precision. However, all types of distress are considered as one class in this study. Neither of the datasets has been made public, so we can only summarize the findings reported. Some researchers combined the two-dimensional (2D) gray-scale images and three-dimensional (3D) laser scanning data and demonstrated the benefit of both, including the 2D and 3D information in the process (*17*). A significant trend observed in the literature is the concentration on

the most common types of pavement distresses, such as longitudinal and transverse cracks. These types of cracks are the most prevalent on a highway network scale, leading to their dominance in datasets. However, this focus creates a bias towards the more frequent crack types and neglects other critical distress types that might be less common. Moreover, models that cannot detect the full range of pavement distress classes have limited practical application.

In the area of deep neural network (DNN)modeling based on the characteristics of pavement cracking, Safaei et al. proposed the segmentation with an adaptive threshold approach, showing promising results in a small setup (130 images) for medium and large cracks (*18*). Nguyen et al. investigate the bigger picture of deep learning methods for crack identification in asphalt pavement (*19*) as they compare the state-of-the-art for crack classification, crack object detection, pixel-level crack segmentation, generative adversarial networks (GANs) for crack segmentation, and crack identification using unsupervised learning. They propose using a high-resolution 2D image dataset with 10,000 images to evaluate the 25 deep-learning methods and found the Faster R-CNN performs best for crack object detection. Cheng et al. show that the improved crack extraction method has an accuracy of the crack classification on a minimal dataset of close to 90% (*20*). Gao et al. propose to leverage low-rank representation in the deep learning pipeline to discriminate images (or frames) with cracks (*21*). They suggest a deep convolutional neural network for crack detection leveraging multilevel features and atrous spatial pyramid pooling (ASPP) (*21*). One study incorporates the latest transformer module with *YOLO* v5, aiming to enhance crack detection by capturing long-range dependencies and learning context information of crack regions. A model trained on a public dataset, including pavement images from India, the Czech Republic, and Japan, achieved F1 scores of 0.6739 and 0.6650 on two online test sets (*22*). Wu et al. introduce a novel detection model based on *YOLO* v5, enhanced with a lightweight crossed feature pyramid network (CFPN) and an improved loss function (*23*). The model was tested on a dataset of 7,076 images covering four common pavement distress types. Results show the model excels in challenging conditions like shadows and overlapping objects, achieving a mean average precision (mAP) of 69.3%.

In summary, most datasets in pavement crack detection studies primarily include standard recurrent distress classes, such as longitudinal and transverse cracks, and the rare classes or more detailed and specific categorizations are often neglected. Most studies focus on improving the overall performance of detection methods rather than addressing particular distress classes, resulting in a lack of sufficient attention to the detection of rare distress types. This imbalance leads to models that are proficient at identifying common distresses but cannot accurately detect and address rarer and critical pavement issues, limiting their practical application. In this study, we propose a novel approach that leverages deep neural networks (DNNs) through a cascade modeling technique combined with low-shot learning, with a particular focus on underrepresented and complex crack types, to enhance the overall effectiveness and practical applicability of pavement crack detection models.

METHODOLOGY

The increasing role of AI for pavement condition assessment requires building a practical yet robust DNN-based distress detection model that considers data in the wild. To this end, we first collect visually different data under different pavement conditions. Second, we define the labels as visually meaningful to both human experts and computer vision algorithms and label the data. Third, we develop a novel object recognition system based on the deep learning You Only Look Once (YOLO) algorithm and propose improvements based on the unique nature of the pavement

imagery.

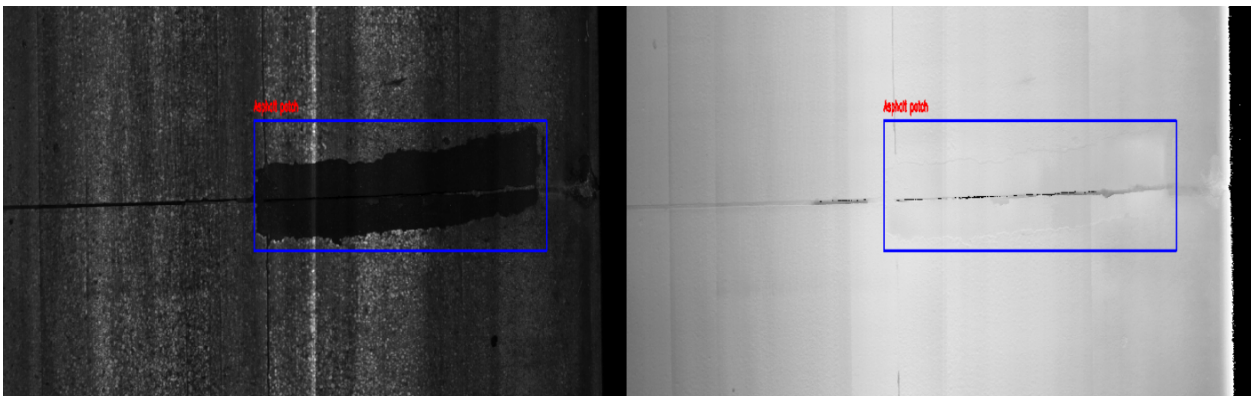


FIGURE 1 A sample of 2D (left) and 3D (right) pavement surface images with an Asphalt Patch annotated.

Step 1: Data in the Wild Acquisition

Pavement surface images used in this study are 2D/3D image data collected in Texas with a specialized 3D laser camera sensor. The 3D laser sensor is mounted on the top back of a van and employs a downward-pointing orientation to project a transverse laser line across the pavement as the van moves. A line camera within the sensor captures the image of this laser line, focusing on the laser's wavelength to minimize ambient light interference. Using the triangulation method, the sensor calculates the relative depth and surface profile of the pavement from the deformation of the laser line on uneven surfaces. This technique allows the construction of a detailed 3D transverse profile of the pavement, significantly enhancing the accuracy and consistency of surface imaging compared to traditional cameras, especially under variable lighting conditions. With high-resolution 2d/3D images, we can focus on addressing the complexities of the crack and pavement conditions, which remain challenging tasks. Each pavement surface image is roughly one lane in width and about 5 meters in length per AASHTO standard recommendations(*24*).

Figure 1 illustrates a sample of the 2D and corresponding 3D images. In the 2D image, each pixel value represents the intensity of the corresponding area on the pavement surface, while in the 3D image, each pixel value indicates the range or relative height. A total of 12 highway sections with Joint Concrete Pavement (JCP) surfaces were selected for this study. Each image has a resolution of 900×1536 pixels, which covers approximately 12 feet in the transverse direction and 6-20 feet in the longitudinal direction, depending on the driving speed.

Step 2: Crack Region Labeling

A total of 8,016 images were collected. These surface images were manually annotated based on the distress classification defined in the Pavement Rater's Manual by the Texas Department of Transportation (TxDOT) and separated into 13 visual classes. Each distress identified is labeled with a bounding box and the corresponding distress class. The bounding box is supposed to cover the extent of a complete cracking within an image, with the background included as tight as possible. The number of annotated instances of distress, the intensity image with the labeled region, and the corresponding 3D laser image representatives are summarized in Table 1 for recurring labels and in Table 2 for rare labels. Deep learning networks require a lot of training data to localize cracks reliably. Thus, we separated our dataset into two classes: the recurring classes and the rare

classes.

TABLE 1 Recurring pavement crack label abbreviation, number of annotated instances of each class in the entire set, TxDOT explanation of the type of the crack, and a 2D (intensity image) and 3D (range image) visual class example.

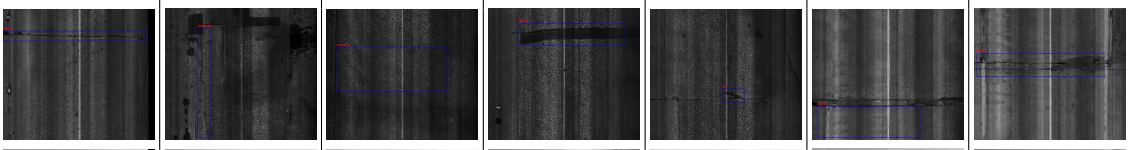
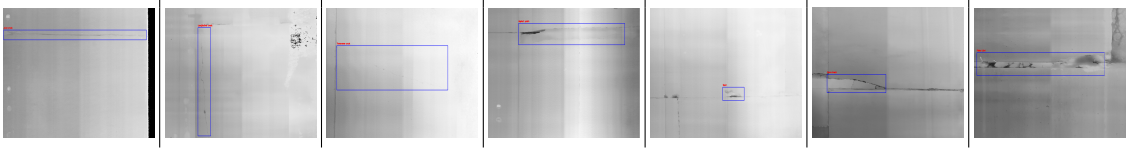
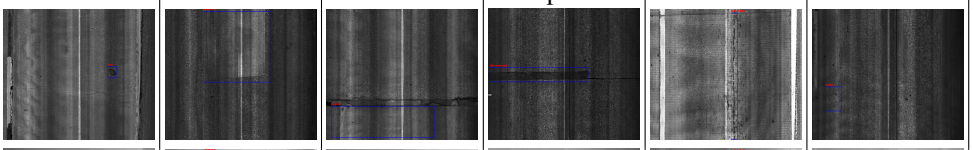
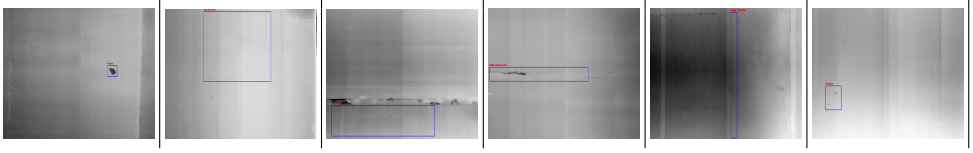
	Recurring classes: number of labeled instances is enough for the robust modeling						
Label →	Joint	L-track	T-crack	A-patch	Spall	C-break	F-joint
Instances	10,286	1512	1,159	602	551	273	243
Description	Joint	Longitudinal crack	Transversal crack	Asphalt Patch	Spall	Corner Break	Failed Joint
Intensity							
Range							

TABLE 2 Rare pavement crack label abbreviation, number of annotated instances of each class in the entire set, TxDOT explanation of the type of the crack, and a 2D (intensity image) and 3D (range image) visual class example.

	Recurring classes: number of labeled instances is enough for the robust modeling **Rare** classes: number of labeled instances is not enough for robust modeling					
Label →	Popout	C-patch	D-crack	FC-patch	S-long	P-out
Instances	190	152	34	27	9	5
Description	Popout	Concrete Patch	D-cracking	Failed concrete patch	Sealed longitudinal	Punchout
Intensity						
Range						

The recurring classes have more than 200 annotated instances in the dataset, as summarized in Table 1, and the rare classes have less than 200 instances (annotated occurrences) in the dataset, as summarized in Table 2.

Step 3: Visual Deep Learning for Crack Localization and Identification

In the third step, we propose three different approaches to modeling: 1. the vanilla deep learning multi-class modeling, 2. the cascade modeling, and 3. binary modeling to extract features followed by traditional machine learning modeling. For the vanilla multi-class, we use YOLOv5 and YOLOv8 (*25*). The YOLO framework (*2*) treats object detection as a single regression problem, and it uses a single convolutional neural network (CNN) to predict bounding boxes and class probabilities simultaneously. The input image is divided into a grid. Each grid cell is responsible for predicting bounding boxes and class probabilities for objects whose center falls within that cell. Each

grid cell predicts a fixed number of bounding boxes. For each bounding box, the network outputs coordinates (center, width, height), confidence scores, and class probabilities. Each grid cell also predicts the probability distribution over all possible object classes for the detected object. After the network outputs the bounding boxes, multiple boxes may overlap. YOLO uses Non-Maximum Suppression to remove redundant boxes and retain only the most accurate ones. YOLOv5 was implemented and published in 2020, and it brought practical improvements, ease of use, and high performance and has become widely used in the community (*25*). YOLOv8 introduced further refinements and optimizations in 2023, focusing on improving accuracy and inference speed and integrating advanced techniques like attention mechanisms and enhanced training strategies (*25*). YOLOv8 incorporates advanced backbone and neck architectures to improve feature extraction and information flow. This includes enhancements in the network's depth and width, as well as improvements in feature pyramid networks or similar components to handle objects at various scales better. YOLOv8 aims to strike a better balance between accuracy and speed. It typically achieves higher precision and recall compared to its predecessors while maintaining or improving inference speed. YOLOv8 uses dropout, batch normalization, and advanced loss functions to prevent overfitting and improve generalization (*25*). For the *cascade* modeling, we combine all annotated instances into one robust crack annotation and train the model as a binary model: crack or no crack. This step has the advantage of removing the black box one-shot decision and providing an analyst with candidates where clear decisions cannot be made. The goal of this step is also to help the system learn in the future, reduce the role of human-in-loop over time, and reduce false positives identified by the system. Then, we only consider regions clearly labeled as generic cracks to classify them into the types of cracks. So, after the binary classifier, we employ another multi-label classifier that only considers regions with detected cracks. Suppose the category does not have enough examples. In that case, we test the following robust pipeline that extracts deep features from binary images in the training data and uses a support vector machines (SVM) traditional classifier to find the related classes. Suppose the crack category has only a few examples. In that case, this will allow us to do domain translation and adapt the model for crack identification for the previously unseen class in the dataset, as we have successfully implemented for different non-consumer tasks in (*26*).

EXPERIMENTS

We measure the effectiveness of the models using Precision (P), Recall (R), F1 measure (F1), Average Precision (AP) per class, and mean Average Precision (mAP) for all classes. This provides us with a set of comprehensive measures of how well a model can detect and localize cracks in images. For each class, a Precision-Recall (PR) curve is plotted. Precision measures the accuracy of the detections (how many of the detected objects are actually relevant), while Recall measures the model's ability to find all relevant objects. The Average Precision (AP) for each class is computed by integrating the Precision-Recall curve. The mean Average Precision (mAP) is calculated by taking the mean of these AP scores across all classes.

Experiment 1. Select Model and Data

We compare the classification performance of two different YOLO versions (5 and 8), two different inputs (Intensity and Two-Channel), and two different classification outcomes (multi-label and binary) to pick the most robust classification baseline. The two-channel input is the 3D scanning detection result (concatenated intensity and range images), and it consistently yields better results. The scanning detection results inform the 2D-3D correspondence for formulating classification

TABLE 3 Table for the average performance of different YOLO Models for all 13 classes. Note that the Precision and Recall are reported for the highest F1.

Method	Intensity				Intensity Binary				Two-Channel				Two-channel Binary			
YOLO	Prec	Rec	mAP	mAP	Prec	Rec	mAP	mAP	Prec	Rec	mAP	mAP	Prec	Rec	mAP	mAP
Model	vision	all	50	50-95	vision	all	50	50-95	vision	all	50	50-95	vision	all	50	50-95
5su train	0.53	0.27	0.24	0.11	0.72	0.66	0.69	0.36	0.45	0.49	0.28	0.14	0.86	0.78	0.75	0.43
8s train	0.54	0.30	0.27	0.12	0.72	0.69	0.69	0.35	0.57	0.40	0.30	0.15	0.87	0.79	0.78	0.45
5su test	0.55	0.24	0.24	0.10	0.75	0.65	0.71	0.36	0.42	0.35	0.29	0.11	0.80	0.68	0.77	0.38
8s test	0.53	0.28	0.25	0.10	0.73	0.65	0.70	0.35	0.48	0.33	0.30	0.13	**0.81**	**0.72**	**0.79**	**0.40**

problems, identifying rare crack types, and informing the final decision, as illustrated in Table 5. Table 5 also shows that a robust binary classifier performs much better than a multi-class one. The explanation can be found in Table 2. For six classes, we do not have enough training data to learn the separation from the related classes. We used this experiment as a guideline to help us decide on the YOLO version for the subsequent ablation study.

TABLE 4 Validation dataset's mAP scores for different IoU and confidence thresholds.

	Prediction IoU Thresholds					
Confidence ↓	0.10	0.20	0.30	0.40	0.50	0.75
0.6000	0.7605	0.7590	0.7587	0.7564	0.7530	0.6255
0.5000	0.8046	0.8025	0.8019	0.7968	0.7901	0.6327
0.4000	0.8293	0.8266	0.8247	0.8177	0.8083	0.6317
0.3000	0.8481	0.8443	0.8415	0.8332	0.8210	0.6285
0.2000	0.8621	0.8577	0.8542	0.8447	0.8309	0.6256
0.1000	0.8727	0.8679	0.8636	0.8528	0.8364	0.6145
0.1500	0.8674	0.8631	0.8590	0.8489	0.8332	0.6209
0.0100	0.8775	0.8704	0.8648	0.8523	0.8293	0.5707

Experiment 2. Ablation study for NMS IoU, Prediction IoU, and Prediction Confidence Thresholding

In this experiment, we report the IoU and Threshold ablations study for the binary YOLOv8 small model on the training data, as it showed the most robust performance in experiment 1. The IoU, or Intersection over Union, is a metric commonly used in visual classification tasks, especially in object detection and image segmentation. IoU is calculated as the ratio of the area of their intersection to the area of their union, where the intersection is the overlapping area between the predicted bounding box and the annotated bounding box, and the union is the total area of the predicted bounding box and the annotated bounding box combined. Thus, the IOU is defined as IoU = Area of Intersection/Area of Union. The IoU score evaluates how well the predicted bounding boxes match the actual bounding boxes. For instance, if an IoU threshold is set, predictions with IoU greater than this threshold are considered true positives, and the smaller ones are ignored. Prior, we found that the high default values of IoU can be very restrictive in non-consumer tasks like crack detection (*27*).

The confidence summarizes the model's score on how the model is confident that the detected object within the bounding box belongs to a particular class. This overall confidence score helps in making a final decision on whether to accept or reject a detected object based on the con-

TABLE 5 Validation dataset's mAP scores for different NMS and prediction IoU thresholds, and confidence threshold 0.001.

	Confidence = 0.001 and Prediction IoU Thresholds						
NMS IoU	0.10	0.20	0.30	0.40	0.50	0.75	AP
0.7500	0.8548	0.8479	0.8423	0.8302	0.8088	0.5471	0.5573
0.7000	0.8648	0.8575	0.8514	0.8387	0.8157	0.5471	0.5594
0.6000	0.8769	0.8690	0.8628	0.8484	0.8226	0.5420	0.5598
0.5000	0.8815	0.8734	0.8666	0.8511	0.8224	0.5392	0.5583
0.4000	0.8835	0.8751	0.8670	0.8479	0.8179	0.5400	0.5571
0.3000	0.8844	0.8742	0.8646	0.8429	0.8122	0.5427	0.5567
0.2000	0.8792	0.8658	0.8548	0.8311	0.8006	0.5437	0.5542
0.1000	0.8641	0.8485	0.8377	0.8159	0.7873	0.5424	0.5499

fidence threshold. In Table 5, we summarize the results of our ablation study on the validation set where we vary the confidence threshold and prediction IoU threshold. Note that mAP scores vary only slightly if the confidence threshold is between 0.01 and 0.2 and IoU is between 0.1 and 0.3. Thus, we adopt the confidence score threshold of 0.01 and IoU threshold of 0.3 as the standard YOLO recommendation for the threshold is 0.01, and it aligns with our experiments. The IoU of 0.3 is still selective enough not to include false positives when the model is evaluated on the unseen dataset. Next, for the confidence 0.01 and varying IoU thresholds in prediction, we test the influence of the NMS IoU setup in Table 5. We find that the NMS IoU of 0.3 in the validation set produces a good balance between false positives and false negatives at each level.

TABLE 6 Performance of the models per recurring label on the test set. The full description of class labels is in Table 1

Recurrent Label	Mean		Joint		L-crack		T-crack		A-patch		Spall		C-break		F-joint	
Method ↓	F1	AP	F1	AP	F1	AP	F1	AP	F1	AP	F1	AP	F1	AP	F1	AP
YOLOv8s	0.552	0.528	0.91	0.96	0.71	0.76	0.66	0.67	0.50	0.51	0.32	0.21	0.44	0.32	0.33	0.27
YOLOv8x	0.538	0.533	0.90	0.95	0.69	0.74	0.65	0.68	0.45	0.49	0.32	0.21	0.37	0.32	0.39	0.34
v8s Bin + SVM	0.502	0.459	0.88	0.92	0.69	0.67	0.62	0.59	0.53	0.50	0.30	0.22	0.26	0.20	0.24	0.11
v8x Bin + SVM	0.465	0.444	0.88	0.92	0.68	0.69	0.62	0.63	0.52	0.48	0.29	0.17	0.23	0.16	0.02	0.05
v8s Cascade	0.476	0.448	0.88	0.92	0.69	0.70	0.62	0.63	0.50	0.46	0.26	0.14	0.24	0.17	0.15	0.12
v8x Cascade	0.489	0.455	0.83	0.85	0.62	0.63	0.61	0.59	0.64	0.60	0.31	0.17	0.32	0.19	0.10	0.15

Experiment 3. Recurring and Rare Class Modeling

In this experiment, we evaluate four different models and their performances for *each* of the class on the test set. We analyze mean average precision (mean AP) for recurring labels in Table 5, and the results paint an interesting picture. The YOLOv8x cascade does not improve F1 and AP for the most frequent classes (Joint, L-crack, and T-crack), and the baseline model produces the best results. The cascade modeling significantly improves A-patch modeling, and the AP jumps from 0.51 to 0.6. D-crack and FC-patch also benefit from the cascade approach as rare labels in Table 5, and the popout is so visually distinctive compared to other classes that the baseline model predicts

better than expected.

TABLE 7 Performance of the model per rare class on the test set. The full description of class labels is in Table 2.

Sparse Label	Mean		C-patch		D-crack		FC-patch		S-long		P-out	
Method ↓	F1.	AP	F1.	AP	F1.	AP	F1.	AP	F1.	AP	F1.	AP
YOLOv8s	0.129	0.145	0.13	0.07	0.00	0.15	0.00	0.02	0.00	0.06	0.52	0.43
YOLOv8x	0.093	0.099	0.13	0.05	0.00	0.07	0.00	0.04	0.00	0.02	0.34	0.31
v8s Binary + SVM	0.107	0.060	0.08	0.04	0.23	0.11	0.00	0.00	0.00	0.00	0.23	0.15
v8x Binary + SVM	0.102	0.068	0.08	0.02	0.13	0.10	0.00	0.00	0.00	0.00	0.29	0.22
v8s Cascade	0.132	0.138	0.10	0.02	0.14	0.08	0.00	0.23	0.00	0.00	0.43	0.36
v8x Cascade	0.156	0.125	0.00	0.01	0.09	0.06	0.25	0.23	0.00	0.00	0.44	0.34

Summary of Findings

In this section, we have conducted three different experiments: the first two were for the parameter selection. At the same time, the third one evaluates the various effects of baseline and three new proposed models for the crack identification task. According to the results of experiment 1, the selection of the baseline model (YOLOv8) and the input (2D/23 combined) are consistent with the findings in the field. The recurrent crack classes with a more significant number of instances were found to produce better performances across the board. Experiment 2 discovered that our models performed better at much lower IoU thresholds, and we selected the NMS IoU of 0.3 and prediction IoU of 0.3 while keeping the confidence level low at 0.001. The experiment highlights the importance of IoU thresholding selection and hyper-parameter choices for NMS. Cascade modeling in Experiment 3 produced a mixed bag of results. For the classes with more than 1000 annotations, it did not improve the performance as it was too restrictive and excluded some true positives, resulting in the drop of the AP. It did improve the performance of the A-patch, D-crack, and FC-patch classifier.

CONCLUSION AND FUTURE WORK

In this paper, we propose MoPaC, a state-of-the-art crack localization and identification framework that effectively recognizes different types of cracks in the pavement from intensity and range images. First, we have made significant improvements in training data set labeling and region annotation and defined categories that are visually meaningful both for human experts and the automated system. We propose the 2D-3D YOLO-based system that robustly detects joints, longitudinal and traversal cracks, as well as asphalt patches captured over larger patches of the road. MoPac's next step is to continue developing high-quality training datasets and increase the number of classes with over 1000 annotations to confirm these initial findings.

ACKNOWLEDGMENTS

The authors would like to acknowledge funding supports received from TxDOT (Research Project #0-7150).

AUTHOR CONTRIBUTIONS

Scouten led the machine learning implementation and analysis, created tables, and contributed to manuscript preparation and editing. Gong led the creation and manual annotation of the dataset, created figures, and contributed to the literature review, manuscript preparation, and editing. Tešić designed the experiments and data analysis and contributed to the literature review and the manuscript preparation and editing. Wang conceived the research idea, supervised the project, and contributed to the manuscript writing and editing.

REFERENCES

1. Gong, H., J. Tešić, J. Tao, X. H. Luo, and F. Wang, Automated Pavement Crack Detection with Deep Learning Methods: What Are the Main Factors and How to Improve the Performance? *Transportation Research Record*, Vol. 2677, No. 10, 2023, pp. 311–323.
2. Redmon, J., S. Divvala, R. Girshick, and A. Farhadi, You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.
3. Redmon, J. and A. Farhadi, YOLO9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7263–7271.
4. Redmon, J. and A. Farhadi, Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.
5. Bochkovskiy, A., C.-Y. Wang, and H.-Y. M. Liao, Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*, 2020.
6. Wang, C.-Y., I.-H. Yeh, and H.-Y. M. Liao, Yolov9: Learning what you want to learn using programmable gradient information. *arXiv preprint arXiv:2402.13616*, 2024.
7. He, K., G. Gkioxari, P. Dollár, and R. Girshick, Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.
8. Cai, Z. and N. Vasconcelos, Cascade R-CNN: High quality object detection and instance segmentation. *IEEE transactions on pattern analysis and machine intelligence*, Vol. 43, No. 5, 2019, pp. 1483–1498.
9. Ronneberger, O., P. Fischer, and T. Brox, U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, Springer, 2015, pp. 234–241.
10. Chen, L.-C., G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, Vol. 40, No. 4, 2017, pp. 834–848.
11. Majidifard, H., P. Jin, Y. Adu-Gyamfi, and W. G. Buttlar, Pavement image datasets: A new benchmark dataset to classify and densify pavement distresses. *Transportation Research Record*, Vol. 2674, No. 2, 2020, pp. 328–339.
12. Arya, D., H. Maeda, S. K. Ghosh, D. Toshniwal, A. Mraz, T. Kashiyama, and Y. Sekimoto, Transfer learning-based road damage detection for multiple countries. *arXiv preprint arXiv:2008.13101*, 2020.
13. Arya, D., H. Maeda, S. K. Ghosh, D. Toshniwal, H. Omata, T. Kashiyama, and Y. Sekimoto, Global road damage detection: State-of-the-art solutions. In *2020 IEEE International Conference on Big Data (Big Data)*, IEEE, 2020, pp. 5533–5539.

14. Mandal, V., L. Uong, and Y. Adu-Gyamfi, Automated road crack detection using deep convolutional neural networks. In *2018 IEEE International Conference on Big Data (Big Data)*, IEEE, 2018, pp. 5212–5215.
15. Ghosh, R. and O. Smadi, Automated Detection and Classification of Pavement Distresses using 3D Pavement Surface Images and Deep Learning. *Transportation Research Record*, Vol. 2675, No. 9, 2021, pp. 1359–1374.
16. Hu, G. X., B. L. Hu, Z. Yang, L. Huang, and P. Li, Pavement Crack Detection Method Based on Deep Learning Models. *Wireless Communications and Mobile Computing*, Vol. 2021, No. 1, 2021.
17. Huang, J., W. Liu, and X. Sun, A Pavement Crack Detection Method Combining 2D with 3D Information Based on Dempster-Shafer Theory. *Computer-Aided Civil and Infrastructure Engineering*, Vol. 29, No. 4, 2014, pp. 299–313.
18. Safaei, N., O. Smadi, A. Masoud, and B. Safaei, An Automatic Image Processing Algorithm Based on Crack Pixel Density for Pavement Crack Detection and Classification. *International Journal of Pavement Research and Technology*, Vol. 15, 2022, pp. 159–172.
19. Son Dong Nguyen, S. D., T. S. Tran, V. P. Tran, H. J. Lee, M. J. Piran, and V. P. Le, Deep Learning-Based Crack Detection: A Survey. *International Journal of Pavement Research and Technology*, Vol. 16, 2022, pp. 943–967.
20. Chang, J., Y. Liu, Z. Shu, H. Zhang, and H. Cao, Pavement Crack Identification Based on Deep Learning and Denoising Model. In *2021 4th International Symposium on Traffic Transportation and Civil Architecture (ISTTCA)*, 2021, pp. 180–185.
21. Gao, Z., X. Zhao, M. Cao, Z. Li, K. Liu, and B. M. Chen, Synergizing Low Rank Representation and Deep Learning for Automatic Pavement Crack Detection. *IEEE Transactions on Intelligent Transportation Systems*, Vol. 24, No. 10, 2023, pp. 10676–10690.
22. Xiang, X., Z. Wang, and Y. Qiao, An improved YOLOv5 crack detection method combined with transformer. *IEEE Sensors Journal*, Vol. 22, No. 14, 2022, pp. 14328–14335.
23. Wu, P., J. Wu, and L. Xie, Pavement distress detection based on improved feature fusion network. *Measurement*, Vol. 236, 2024, p. 115119.
24. AASHTO, Standard Specification for File Format of Two-Dimensional and Three-Dimensional (2D/3D) Pavement Image Data. Washington, D.C., 2023.
25. Jocher, G., A. Chaurasia, and J. Qiu, *Ultralytics YOLO*, 2023.
26. *David Heyse*, *Nicholas Warren*, and **Jelena Tešić**, Identifying maritime vessels at multiple levels of descriptions using deep features. In *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications*, SPIE, 2019, Vol. 11006, pp. 423 – 431.
27. Warren, N., B. Garrard, S. E., and Tešić, J., Transfer Learning of Deep Neural Networks for Visual Collaborative Maritime Asset Identification. In *2018 IEEE 4th International Conference on Collaboration and Internet Computing (CIC)*, 2018, pp. 246–255.