Yet Another Deep Learning Approach for Road Damage Detection using Ensemble Learning

Vinuta Hegde*†, Dweep Trivedi*†, Abdullah Alfarrarjeh‡, Aditi Deepak†, Seon Ho Kim†, Cyrus Shahabi†
† *Integrated Media Systems Center, University of Southern California, Los Angeles, CA 90089, USA*
‡ *Department of Computer Science, German Jordanian University, Amman, Jordan*
⟨ vinutahe, dtrivedi, adeepak, seonkim, shahabi ⟩@usc.edu, ⟨ abdullah.alfarrarjeh ⟩@gju.edu.jo

***Abstract*—For efficient road maintenance, an automated monitoring system is required to avoid laboriously and time-consuming manual inspection by road administration crews. One potential solution is to utilize image processing-based technologies, especially, as various sources of images have readily been available, e.g., surveillance cameras, in-vehicle cameras, or smartphones. Such image-based solutions enable detecting and classifying road damages. This paper introduces deep learning-based image analysis for road damage detection and classification. Our ensemble learning approaches with test time augmentation were thoroughly evaluated using the 2020 IEEE Big Data Global Road Damage Detection Challenge Dataset. Experimental results show that our approaches achieved an F1 score of up to 0.67, allowing us to win the Challenge.**

***Index Terms*—Deep Learning, Road Damage Detection and Classification, Object Detection, Ensemble Learning, Urban Street Analysis**

I. Introduction

Road networks activate economic and social development (e.g., trade, tourism, education, employment, health). Therefore, governments dedicate large budgets annually to construct and maintain road networks to facilitate transportation throughout different areas of cities and across cities. For example, in 2020, the United States has allocated $5 billion to construct new roads and maintain the existing ones [1].

Given that roads are subjective to damages due to various reasons such as weather, aging, and traffic accidents, governments need road inspection systems to evaluate the road surface conditions regularly. One of these systems involves human intervention where a specialized crew conduct site inspection visit to rate road conditions, but it is time-consuming and laborious. Therefore, other automated monitoring systems have been devised, including vibration-based [2], laser-scanning-based [3], and image-based [4]–[7] methods. Each of these methods has its own limitations. Vibration and laser-scanning methods need special equipment, so they are expensive. Moreover, they require road closures during an inspection. Hence, despite their high accuracy, they are not very practical at scale and not preferred. On the other hand, image-based methods are inexpensive and do not need physical existence on the roads[1]; nonetheless, they have been less accurate than vibration or laser-scanning methods.

A critical recent technical trend in image analysis is the use of convolutional neural networks, which significantly improves image-based analysis accuracy. Such methods have been utilized in various smart city applications such as street cleanliness classification [14], [15], material recognition [16], [17], situation awareness of disasters [18], traffic flow analysis [19], and image search [20], [21]. Thus, several image-based methods have naturally been proposed in the domain of road damage detection too. In general, these methods are categorized into two groups. One group of methods focused on detecting road damages without providing the details about damage types [4]. The other group focused on both road damage detection and classification of damage types. For example, Zhang *et al.* [5] and Akarsu *et al.* [6] proposed approaches to detect directional cracks. Recently, Maeda *et al.* [7] proposed a classification of eight road damage types and collected a dataset of street images captured in Japan and annotated with the damage types. This dataset was released to the public and used for the 2018 IEEE Big Data Road Damage Detection Challenge; hence, several research teams provided different technical solutions for the problem [22]–[26]. Another image dataset was recently released for the 2020 IEEE Big Data Global Road Damage Detection Challenge [27]. This recent dataset consists of images labeled with the same damage types provided by Maeda's dataset; however, it includes more images collected from three countries: Czech, India, and Japan.

To address the 2020 IEEE Big Data Global Road Damage Detection Challenge, we first investigated several state-of-the-art object detection algorithms and applied them for road damage detection. To further improve the accuracy of the trained models generated by object detection algorithms, we devised three ensemble learning approaches: a) the Ensemble Prediction approach (EP) which applies an ensemble of the predictions obtained from images generated by the test time augmentation (TTA) procedure, b) the Ensembled Model approach (EM) which uses multiple trained models

*These authors contributed equally to this work.

[1]Spatial crowdsourcing mechanisms can be used for collecting images for different geographical locations by the public [8], [9]. Moreover, since image are usually tagged with their locations (if not, they can be localized [10]), measuring the visual coverage of a location helps to determining the locations which needs crowdsourcing [11]–[13].

978-1-7281-6251-5/20/$31.00 ©2020 IEEE

for prediction, and c) a hybrid approach (EM+EP) which uses an ensembled model from EM for generating predictions for the images generated by the TTA procedure. Then, a thorough evaluation was conducted using the public dataset of the 2020 IEEE Big Data Global Road Damage Detection Challenge. Our evaluation presents the trade off between speed and accuracy for all variants of the trained models. The source code of our solution is available at (https://github.com/USC-InfoLab/rddc2020).

The remainder of this paper is organized as follows. Section II introduces the classification of road damages, and presents our solution. In Section III we report our experimental results. Finally, in Section V, we conclude.

II. Road Damage Detection Approach

A. Image Dataset

The image dataset provided by the IEEE Big Data 2020 Global Road Damage Detection Challenge was collected from three countries: Czech Republic (CZ), India (IN), and Japan (JP). The training dataset is composed of 21,041 images, and each image is annotated by one or more classes of road damages; specifically, the classes of road damages (see Table I) are based on the classification proposed by Maeda *et al.* [7] (which is adopted by the Japan Road Association [28]). Examples of these classes of road damage types are shown in Figs. 1a-1h. Some of the training images are not annotated with any road damage class; thus they are free from any road damages. In this paper, following the challenge guidelines, we considered only the images annotated by four specific classes of road damages (namely, $D00$, $D10$, $D20$, and $D40$) and the remaining images were considered as free of road damages when designing our solution.

B. Proposed Approach

With the recent advancements in image content analysis using convolutional neural networks (CNN), several CNN-based methods have been proposed for object detection. Generally, CNN based object detection algorithms can be divided in two categories [29]: one-stage detectors and two-stage detectors. Two-stage detectors such as R-CNN (Regions with CNN) introduced by Girshick *et al.* [30] are composed of two steps: object region proposal followed by region classification. The first step exhaustively searches for regions containing potential objects. Then, these regions are classified using a CNN classifier to predict the existence of an object. However, R-CNN has shown performance limitations so was extended to different variants, including Fast R-CNN [31] and Faster R-CNN [32]. The one-stage detectors propose bounding boxes directly from input images without object region proposal step, making them time efficient and useful for real-time devices. One of the well-known one-stage detectors is "You Only Look Once" (YOLO) [33]. Several versions of YOLO have been proposed, including YOLOv3 [34], YOLOv4 [35], and ultralytics-YOLO (u-YOLO) [36]. Using the given dataset in the challenge, we first evaluated several CNN-based object detection methods including, Faster R-CNN, YOLOv3, and u-YOLO. We have found that u-YOLO outperformed both Faster R-CNN and YOLOv3. Therefore, we adopted u-YOLO as the core of our proposed approaches.

To improve the robustness of the u-YOLO method, we utilize the test time augmentation (TTA) procedure. TTA applies several transformations (e.g., horizontal flipping, increasing image resolution) to a test image. Then, predictions are generated for all the images individually (i.e., the original image and the augmented ones) using the same trained u-YOLO model. Subsequently, the predictions for all images are combined into a single list and the non-maximum suppression (NMS) algorithm is employed to generate the final output. NMS aims at filtering the overlapped or duplicate proposed predictions from the combined list. This approach is referred to as *Ensemble Prediction* (EP) (see Fig. 2). Another approach is to ensemble different variants of u-YOLO models. Given that training a u-YOLO model involves tuning different hyper-parameters, using different combinations of these parameters generates different trained models. A subset of these models is selected such that they maximize the overall accuracy. Each image is passed through all the selected models and predictions from each model are averaged before applying NMS. This approach is referred to as *Ensemble Model* (EM) (see Fig. 3). As an ensemble technique reduces the prediction variance, a better accuracy can be achieved. The last approach is to extend EM with the TTA procedure used in EP. This approach simply applies the EP approach to each model in EM. In particular, after transforming a test image using TTA, the augmented images are fed into each model of EM. Then, the predicted list of bounding boxes from the augmented images for each model are averaged before applying NMS. This latest approach is referred to as *Ensemble Model with Ensemble Prediction* (EM+EP) (see Fig. 4).

III. Experiments

A. Dataset and Settings

The image dataset provided by the IEEE Big Data 2020 Road Damage Detection Challenge has been used for training purposes (referred to as $\mathcal{D}$). Following the challenge guidelines, we considered only the images annotated by four specific classes of road damages (namely, $D00$, $D10$, $D20$, and $D40$) and the remaining images were considered as free of road damages when training our models. Figs. 7b and 7c show the image distribution of the dataset with respect to damage vs. no-damage and road damages classes, respectively. Since some road damage classes have small number of images, especially the ones from Czech and India, we augmented $\mathcal{D}$ with synthesized images. Such images are generated using the processing methods provided by the Python Augmentor library [37]. While using the Augmentor tool, four types of image processing methods (i.e., sharpen, multiply, additive Gaussian noise, and affine texture mapping) were used to

TABLE I: Road Damage Types [7]

Damage Type			Detail	Class Name
Crack	Linear Crack	Longitudinal	Wheel mark part	$D00$
			Construction joint part	$D01$
		Lateral	Equal interval	$D10$
			Construction joint part	$D11$
	Alligator Crack		Partial pavement, overall pavement	$D20$
Other Corruption			Rutting, bump, pothole, separation	$D40$
			Crosswalk blur	$D43$
			White/Yellow line blur	$D44$

(a) $D00$ (b) $D01$ (c) $D10$ (d) $D11$

(e) $D20$ (f) $D40$ (g) $D43$ (h) $D44$

Fig. 1: Image Examples of the Road Damage Classes [27]

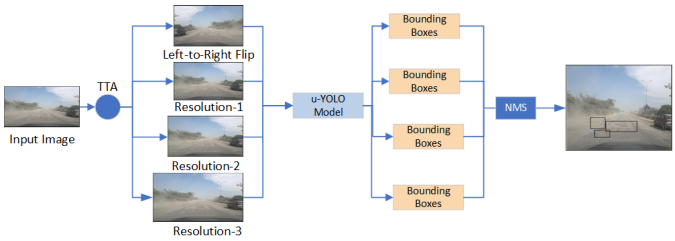


Fig. 2: The Ensemble Prediction (EP) Approach using the Test Time Augmentation Procedure

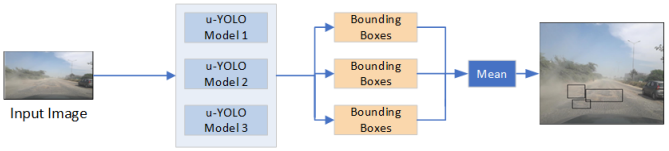


Fig. 3: The Ensemble Model Approach (EM) using Multiple Variants of u-YOLO Model

assure that the road damage scenes were not affected[2]. The use

[2]Some image processing techniques, such as rotating, may lead to a confusion (e.g., vertical vs. horizontal cracks) among road damage types.

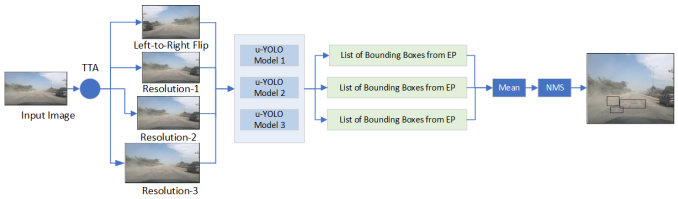


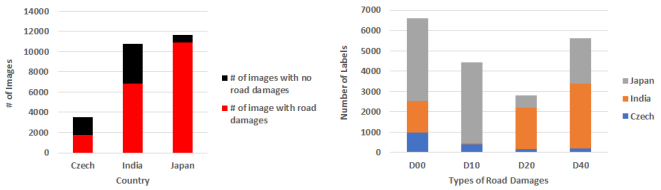
Fig. 4: The Ensemble Model with Ensemble Prediction Approach (EM+EP) using the Test Time Augmentation Procedure and Multiple Variants of u-YOLO Model

TABLE II: Damage annotation distribution

Class Name	Original Dataset ($\mathcal{D}$)			Augmented Dataset ($\mathcal{D}_a$)		
	Czech	India	Japan	Czech	India	Japan
$D00$	988	1555	4049	1741	2218	4049
$D10$	399	68	3979	965	339	3979
$D20$	161	2021	6199	595	2544	6199
$D40$	197	3187	2243	654	4200	2243

of the augmentation technique resulted in a new augmented training dataset (referred to $\mathcal{D}_a$) and its distribution compared with $\mathcal{D}$ is illustrated in Table II (the classes which were affected by augmentation are highlighted in gray).

Different pre-trained object detection models were fine-



(a) w.r.t. damage vs. no damage (b) w.r.t. Road Damage Classes

Fig. 5: Image Dataset Distribution

TABLE III: Variants of the Trained Models Used for the EM Approach

Model	Dataset	Image Size	Batch	Optimizer
u-YOLO model #1	$\mathcal{D}$	640	16	SGD
u-YOLO model #2	$\mathcal{D}_a$	448	32	SGD
u-YOLO model #3	$\mathcal{D}_a$	640	32	SGD

tuned based on either $\mathcal{D}$ or $\mathcal{D}_a$. While fine-tuning, several hyperparameters were explored to generate a better version of the pre-trained model, including image size (e.g., 448 or 640), optimizer (Adam or stochastic gradient descent (SGD)), batch size (e.g., 16 or 32), and number of epochs. During the training phase, snapshots of the trained model were saved after every five epochs. Regarding the EM approach, we used the models shown in table III where these trained models were varied based on different hyper parameters while training. Regarding the EP approach, each test image was transformed into multiple images using left-right flip, and scaling with a ratio of 0.67 and 0.83.

For testing purposes, the organizers of the challenge have provided another dataset of 2,631 images (referred to $\mathcal{D}_T$)[3]. To improve the accuracy of prediction, we explored two hyperparameters during the testing phase: minimum confidence threshold ($\mathbb{C}$) and non-maximum suppression (NMS). $\mathbb{C}$ was used to discard the predicted bounding boxes whose confidence score were less than $\mathbb{C}$. Meanwhile, NMS was used to discard some of the overlapped predicted boxes. The used values of these two parameters are listed in Table IV.

In what follows, we present the evaluation results in terms of accuracy and performance. The accuracy of the approaches is reported using F1 score[4] while the performance is measured using detection/inference time.

B. Evaluation Results

Initially, various object detection algorithms were used to generate different trained models based on $\mathcal{D}$. In particular, we used Faster-RCNN, YOLOv3, and u-YOLO and the generated models were evaluated using $\mathcal{D}_T$ and their F1 scores are reported in Table V. Since u-YOLO achieved the highest F1 score, we chose it for further evaluation by varying the values of the hyperparameters and by using the augmented training dataset (i.e., $\mathcal{D}_a$).

[3]Actually, the organizers have provided two testing datasets for two different test rounds (test1 is composed of 2,631 images while the test2 is composed of 2,664 images). Throughout the paper, we report the results using the test1 dataset.

[4]Since the provided test images do not have ground-truth boxes, we only reported F1 scores that were calculated using the website of the road damage detection challenge (https://rdd2020.sekilab.global/).

TABLE IV: Parameter Values for Experiments

Parameter	Values
$\mathbb{C}$	0.01, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3
NMS	0.5, 0.6, 0.7, 0.8, 0.9, 0.99, 0.999

TABLE V: F1-Scores of Trained Models using Various Object Detection Algorithms

Model	F1-Score
Faster R-CNN [32]	0.508353
YOLOV3 [34]	0.505320
u-YOLO [36]	**0.63024**

In the testing phase, the u-YOLO model was evaluated exhaustively using various values of the $\mathbb{C}$ and NMS parameters as shown in Table VI. To determine the best values for both $\mathbb{C}$ and NMS, we used the grid search mechanism. Experimentally, the u-YOLO model achieved the highest F1 score (i.e., 0.63) when $\mathbb{C}$ = 0.15 and NMS = 0.999.

To evaluate the impact of the augmentation technique, the u-YOLO model was trained using $\mathcal{D}_a$ and compared with the model trained on $\mathcal{D}$. Table VII shows the F1 scores of this model by varying the values of $\mathbb{C}$ and NMS. The u-YOLO model based on $\mathcal{D}_a$ gained improvement in F1 score reaching up 0.62. As u-YOLO applies feature augmentation during training time by default, using $\mathcal{D}_a$ was not effective to boost the F1 score.

(a) u-YOLO (b) u-YOLO w/ EP

Fig. 6: Detection Results using u-YOLO vs. u-YOLO w/ EP

Our ensemble learning approaches (i.e., EP, EM, and EM+EP) enhanced the accuracy of the u-YOLO model. We evaluated these approaches in terms of accuracy and performance as shown in Table VIII and Table IX, respectively. The EP approach achieved a significant improvement in accuracy compared to u-YOLO by obtaining an F1 score of 0.66. Since the YOLO algorithm uses a predefined set of boxes (referred to as anchor boxes in [38]) to predict bounding boxes, EP helps to reduce false negatives by running inference on images with different resolution and angles (as shown in Fig. 6). However, this accuracy improvement comes at the cost of performance. The increase in prediction time in the

TABLE VI: F1 Scores of the u-YOLO Model trained on $\mathcal{D}$, varying $\mathbb{C}$ and NMS

NMS	$\mathbb{C}$						
	0.01	**0.05**	**0.1**	**0.15**	**0.2**	**0.25**	**0.3**
0.5	0.25631	0.43630	0.51008	0.54420	0.56031	0.56477	0.56495
0.6	0.28401	0.44353	0.51397	0.54520	0.56037	0.56433	0.56448
0.7	0.32749	0.46876	0.52286	0.54862	0.56227	0.56609	0.56455
0.8	0.36554	0.51384	0.56174	0.57528	0.57679	0.57412	0.56856
0.9	0.40922	0.55150	0.60105	0.61592	0.57679	0.60825	0.59284
0.99	0.43172	0.56329	0.61368	0.62840	0.62883	0.61973	0.60330
0.999	0.44032	0.56807	0.61659	**0.63024**	0.62989	0.62007	0.60378

TABLE VII: F1 Scores of the u-YOLO Model trained on $\mathcal{D}_a$, varying $\mathbb{C}$ and NMS

NMS	$\mathbb{C}$						
	0.01	**0.05**	**0.1**	**0.15**	**0.2**	**0.25**	**0.3**
0.5	0.27873	0.45273	0.51756	0.55364	0.56593	0.56686	0.56132
0.6	0.30299	0.45786	0.51940	0.55430	0.56605	0.56723	0.56148
0.7	0.34873	0.48069	0.52710	0.55606	0.56602	0.56657	0.56137
0.8	0.38821	0.52022	0.55912	0.57841	0.57933	0.57523	0.56527
0.9	0.42673	0.55732	0.59742	0.61292	0.61338	0.60215	0.58751
0.99	0.44623	0.56890	0.60803	0.62286	0.62239	0.61267	0.59741
0.999	0.45519	0.57458	0.61209	**0.62591**	0.62437	0.61370	0.59800

TABLE VIII: F1 Scores of the variants of the u-YOLO model

Model Name	F1-Score
u-YOLO	0.63024
u-YOLO with EP	0.66697
u-YOLO with EM	0.646515
u-YOLO with EM+EP	**0.676181**

TABLE IX: Detection Performance of the variants of the u-YOLO model

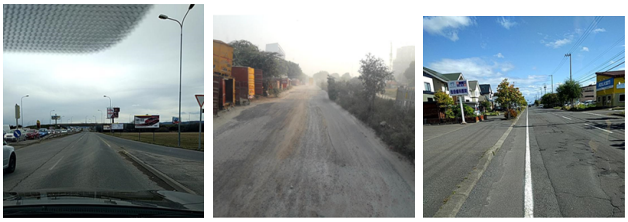
Approach	Detection time per image (sec)	# of images per second
u-YOLO	0.04167	23.99
u-YOLO with EP	0.10867	9.20
u-YOLO with EM	0.118842	8.42
u-YOLO with EM+EP	0.32498	3.08

EP approach can be attributed to the image augmentation process and performing the detection process on all the augmented images. The EM approach increased u-YOLO's F1 score from 0.63 to 0.64. Since EM relies on a group of trained models for final prediction, it reduces the influence of having an over-fitted single model, which helps in improving the prediction accuracy. Though EM and EP have similar detection performance (see Table IX), the detection time using EM increases linearly with the number of models used in ensemble and the detection time using EP increases linearly with the number of augmented images generated by TTA. The hybrid ensemble approach (i.e., EM+EP) achieved the highest F1-score reaching up to 0.67; however it provided the slowest performance as this approach involves generating the TTA images as well as prediction using all models in EM [5].

IV. Observations

In what follows, we report some approaches which we attempted; however, note that these directions have not obtained better accuracy results.

- Given that $\mathcal{D}$ was collected from different countries, we attempted generating a local model per country similar to Arya *et al.* [27]. The motivation of this approach was that we noticed significant differences in the street views from different countries, especially the images collected from India (see Fig. 7). Thus, we trained a detection model per country. However, the Czech and India models ended up over-fitting due to the small number of training images.
- We tried training individual models for India and Czech by borrowing images from Japan. This trial was based on the observation that the images of these countries contain too small number of images for some classes. Even though this might have created a robust model, it did not improve the overall accuracy significantly. However, creating one model for the entire dataset $\mathcal{D}$ enables learning more robust features compared to learning one model per country.



(a) Czech Republic (b) India (c) Japan

Fig. 7: Example Images of $\mathcal{D}$ Collected from Three Countries

V. Conclusion

An automated solution for road damage detection and classification using image analysis is nowadays timely needed for smart city applications. In this paper, we designed deep learning approaches based on one of the state-of-the-art object

[5]We also refer the reader to the paper summarizing the 2020 IEEE Big Data Global Road Damage Detection Challenge and the proposed solutions by other participants [39].

detection approaches, namely YOLO. To increase the accuracy of the trained models generated by YOLO, we presented three approaches using ensemble learning. One approach uses multiple transformed images of the test image to ensemble the final output. The other approach ensembles multiple trained models and averages predictions from these trained models. The third approach combines the latest two approaches. All these three approaches were evaluated using the public image dataset provided by the 2020 IEEE Big Data Global Road Damage Detection Challenge. Our approaches were able to achieve an F1 score of up to 0.67. As part of future work, we plan to integrate this solution in our visionary framework (Translational Visual Data Platform, TVDP [40]) for smart cities.

ACKNOWLEDGMENT

This research has been supported in part by the USC Integrated Media Systems Center and unrestricted cash gifts from Oracle.

REFERENCES

[1] E. Chao, "US department of transportation - budget highlights 2020," 2020. [Online]. Available: https://bit.ly/3kn5Mtu

[2] B. X. Yu and X. Yu, "Vibration-based system for pavement condition evaluation," in *AATT*, 2006, pp. 183–189.

[3] Q. Li, M. Yao, X. Yao, and B. Xu, "A real-time 3D scanning system for pavement distortion inspection," *MST*, vol. 21, no. 1, p. 015702, 2009.

[4] A. Zhang, K. C. Wang, B. Li, E. Yang, X. Dai, Y. Peng, Y. Fei, Y. Liu, J. Q. Li, and C. Chen, "Automated pixel-level pavement crack detection on 3d asphalt surfaces using a deep-learning network," *CACAIE*, vol. 32, no. 10, pp. 805–819, 2017.

[5] L. Zhang, F. Yang, Y. D. Zhang, and Y. J. Zhu, "Road crack detection using deep convolutional neural network," in *ICIP*. IEEE, 2016, pp. 3708–3712.

[6] B. Akarsu, M. KARAKÖSE, K. PARLAK, A. Erhan, and A. SARIMADEN, "A fast and adaptive road defect detection approach using computer vision with real time implementation," *IJAMEC*, vol. 4, no. Special Issue-1, pp. 290–295, 2016.

[7] H. Maeda, Y. Sekimoto, T. Seto, T. Kashiyama, and H. Omata, "Road damage detection and classification using deep neural networks with smartphone images," *CACAIE*.

[8] L. Kazemi and C. Shahabi, "Geocrowd: enabling query answering with spatial crowdsourcing," in *SIGSPATIAL GIS*. ACM, 2012, pp. 189–198.

[9] A. Alfarrarjeh, T. Emrich, and C. Shahabi, "Scalable spatial crowdsourcing: A study of distributed algorithms," in *MDM*, vol. 1. IEEE, 2015, pp. 134–144.

[10] A. Alfarrarjeh, S. H. Kim, S. Rajan, A. Deshmukh, and C. Shahabi, "A data-centric approach for image scene localization," in *Big Data*. IEEE, 2018, pp. 594–603.

[11] A. Alfarrarjeh, S. H. Kim, A. Deshmukh, S. Rajan, Y. Lu, and C. Shahabi, "Spatial coverage measurement of geo-tagged visual data: A database approach," in *BigMM*. IEEE, 2018, pp. 1–8.

[12] A. Alfarrarjeh, Z. Ma, S. H. Kim, and C. Shahabi, "3D spatial coverage measurement of aerial images," in *MMM*. Springer, 2020, pp. 365–377.

[13] A. Alfarrarjeh, Z. Ma, S. H. Kim, Y. Park, and C. Shahabi, "A web-based visualization tool for 3D spatial coverage measurement of aerial images," in *MMM*. Springer, 2020, pp. 715–721.

[14] H. Begur, M. Dhawade, N. Gaur, P. Dureja, J. Gao, M. Mahmoud, J. Huang, S. Chen, and X. Ding, "An edge-based smart mobile service system for illegal dumping detection and monitoring in San Jose," in *UIC*. IEEE, 2017, pp. 1–6.

[15] A. Alfarrarjeh, S. H. Kim, S. Agrawal, M. Ashok, S. Y. Kim, and C. Shahabi, "Image classification to determine the level of street cleanliness: A case study," in *BigMM*. IEEE, 2018.

[16] S. Bell, P. Upchurch, N. Snavely, and K. Bala, "Material recognition in the wild with the materials in context database," in *CVPR*, 2015, pp. 3479–3487.

[17] A. Alfarrarjeh, D. Trivedi, S. H. Kim, H. Park, C. Huang, and C. Shahabi, "Recognizing material of a covered object: A case study with graffiti," in *ICIP*. IEEE, 2019, pp. 2491–2495.

[18] A. Alfarrarjeh, S. Agrawal, S. H. Kim, and C. Shahabi, "Geo-spatial multimedia sentiment analysis in disasters," in *DSAA*. IEEE, 2017, pp. 193–202.

[19] S. H. Kim, J. Shi, A. Alfarrarjeh, D. Xu, Y. Tan, and C. Shahabi, "Real-time traffic video analysis using intel viewmont coprocessor," in *DNIS*. Springer, 2013, pp. 150–160.

[20] A. Alfarrarjeh, C. Shahabi, and S. H. Kim, "Hybrid indexes for spatial-visual search," in *ACM MM Thematic Workshops*. ACM, 2017, pp. 75–83.

[21] A. Alfarrarjeh, S. H. Kim, V. Hegde, C. Shahabi, Q. Xie, S. Ravada *et al.*, "A class of R*-tree indexes for spatial-visual search of geo-tagged street images," in *ICDE*. IEEE, 2020, pp. 1990–1993.

[22] Y. J. Wang, M. Ding, S. Kan, S. Zhang, and C. Lu, "Deep proposal and detection networks for road damage detection and classification," in *Big Data*. IEEE, 2018, pp. 5224–5227.

[23] W. Wang, B. Wu, S. Yang, and Z. Wang, "Road damage detection and classification with faster R-CNN," in *Big Data*. IEEE, 2018, pp. 5220–5223.

[24] A. Alfarrarjeh, D. Trivedi, S. H. Kim, and C. Shahabi, "A deep learning approach for road damage detection from smartphone images," in *Big Data*. IEEE, 2018, pp. 5201–5204.

[25] L. Ale, N. Zhang, and L. Li, "Road damage detection using retinanet," in *Big Data*. IEEE, 2018, pp. 5197–5200.

[26] R. Manikandan, S. Kumar, and S. Mohan, "Varying adaptive ensemble of deep detectors for road damage detection," in *Big Data*. IEEE, 2018, pp. 5216–5219.

[27] D. Arya, 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.

[28] *Maintenance and Repair Guide Book of the Pavement 2013*, 1st ed. Tokyo, Japan: Japan Road Association, 04 2017.

[29] L. Jiao, F. Zhang, F. Liu, S. Yang, L. Li, Z. Feng, and R. Qu, "A survey of deep learning-based object detection," *IEEE Access*, vol. 7, pp. 128 837–128 868, 2019.

[30] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *CVPR*, 2014, pp. 580–587.

[31] R. Girshick, "Fast R-CNN," in *ICCV*, 2015, pp. 1440–1448.

[32] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *NIPS*, 2015, pp. 91–99.

[33] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *CVPR*, 2016, pp. 779–788.

[34] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," *arXiv*, 2018.

[35] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," *arXiv preprint arXiv:2004.10934*, 2020.

[36] "YOLOv5," 2020. [Online]. Available: https://github.com/ultralytics/yolov5

[37] M. Bloice, "Python augmentor tool," 2016. [Online]. Available: https://augmentor.readthedocs.io/en/master/

[38] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," in *CVPR*, 2017, pp. 7263–7271.

[39] D. Arya, H. Maeda, S. K. Ghosh, D. Toshniwal, H. Omata, T. Kashiyama, and Y. Sekimoto, "Global road damage detection: State-of-the-art solutions," 2020.

[40] S. H. Kim, A. Alfarrarjeh, G. Constantinou, and C. Shahabi, "TVDP: Translational visual data platform for smart cities," in *ICDEW*. IEEE, 2019, pp. 45–52.