Advances in Bridge Engineering

ORIGINAL INNOVATION

Open Access

Advanced bridge visual inspection using real-time machine learning in edge devices

Mahta Zakaria, Enes Karaaslan and F. Necati Catbas*

*Correspondence: catbas@ucf.edu

Department of Civil, Environmental and Construction Engineering, University of Central Florida, Orlando, Florida 32816, USA

Abstract

Conventional methods for bridge inspection are labor intensive and highly subjective. This study introduces an optimized approach using real-time learning-based computer vision algorithms on edge devices to assist inspectors in localizing and quantifying concrete surface defects. To facilitate a better AI-human interaction, localization and quantification are separated in this study. Two separate learning-based computer vision models are selected for this purpose. The models are chosen from several available deep learning models based on their accuracy, inference speed, and memory size. For defect localization, Yolov5s shows the most promising results when compared to several other Convolutional Neural Network architectures, including EfficientDet-d0. For the defect quantification model, 12 different architectures were trained and compared. UNet with EfficientNet-b0 backbone was found to be the best performing model in terms of inference speed and accuracy. The performance of the selected model is tested on multiple edge-computing devices to evaluate its performance in real-time. This showed how different model quantization methods are considered for different edge computing devices. The proposed approach eliminates the subjectivity of human inspection and reduces labor time. It also guarantees human-verified results, generates more annotated data for AI training, and eliminates the need for post-processing. In summary, this paper introduces a novel and efficient visual inspection methodology that uses a learning-based computer vision algorithm optimized for real-time operation in edge devices (i.e., wearable devices, smartphones etc.).

1 Introduction

The need for timely and accurate inspections of structures is growing as America's infrastructure ages. According to a 2022 study from the American Road Transportation Builders Association (ARTBA), an analysis of the recently released US Department of Transportation 2021 National Bridge Inventory (NBI) database reveals that 47,052 bridges are classified as structurally deficient and in poor condition (ARTBA, 2022). Structural Health Monitoring (SHM) is a data-driven method to ensure the safety of infrastructure. In SHM at the local level (LSHM), also known as routine inspection, inspectors search for defects on the surface of the structure. Conventional bridge inspection methods are conducted using basic handheld tools and measuring devices. Oftentimes, structures have locations that are hard to reach, or traffic restricts operations. Inspectors are required to physically inspect all surfaces of the

structure, locating and measuring the defects is a time-consuming process, especially conducted using a visual guide (Washer false, 2019). This oftentimes also requires road closures. On the other hand, the inspection reports are sometimes inconsistent with each other, and the inconsistency increases with more severe defects (Washer false, 2019).

The stated obstacles in visual inspections can be better managed using novel technologies. With the advances in computer vision and robotic technologies, more effective methods are being introduced to accelerate inspection times, increase data accuracy, facilitate access to hard-to-reach locations, eliminate the need for road closures, and provide impartial results. SHM at the local level can significantly benefit from computer vision algorithms. The non-contact, low-cost, and accurate vision sensors along with advancements in computer vision techniques, especially machine-learning based algorithms, have made breakthroughs in visual inspection of structures (Dong & Catbas, 2021). The use of robotic instruments and Unmanned Aerial Vehicles (UAVs) have also improved efficiency by eliminating human interference and allowing better access to hard-to-reach locations (Seo false, 2018). While the use of vision sensors and UAVs have significant benefits for visual data collection, analysis and processing of the collected data is usually conducted offsite, either by human inspectors, or computer vision algorithms. Post-processing is also a labor-intensive task, which consumes a significant amount of time and can be computationally expensive. Moreover, the quality of data may easily be impacted by ambient factors, such as vibration in collected footage due to wind or dark imagery due to low illumination. Hence, data collection is often repeated, resulting in multiple visits to the site, consequently invalidating the efficiency of the method. Even with meticulous planning of UAV flight plans, the collected data for local defects needs to be verified by the inspector which adds to post-processing labor. Other methods of data collection, including handheld cameras or vehicles, follow a similar process. In all cases, data is saved on a local drive or in a cloud and processed offsite.

Advances in machine learning based computer vision algorithms and edge-computing devices may potentially lead to end-to-end visual inspection practices. Instead of post-processing collected inspection data, a highly optimized machine learning system can detect structural defects in real-time, quantify the defect regions, and analyze them for condition assessment with reasonable accuracy.

This paper introduces a novel visual inspection methodology that uses a learning-based computer vision algorithm optimized for real-time operation in edge devices (i.e., wearable devices, smartphones etc.). The explored system aims to assist the inspector by accelerating certain routine tasks such as localizing and measuring surface defects (i.e., cracks, spalling). In this method, the human inspector can interact with the Artificial Intelligence (AI) system through the human-machine interface of the edge device. The proposed methodology is designed to be deployed in edge computing devices that facilitate human-computer interaction such as Augmented Reality (AR) or Mixed Reality (MR) platforms. To investigate the system's effectiveness, a wearable holographic see-through headset was chosen for a case study.

The interactive nature of this methodology provides results that are verified by professional inspectors. With each inspection, the methodology generates more annotated human-verified data that can be used for improving the accuracy of the AI system. This

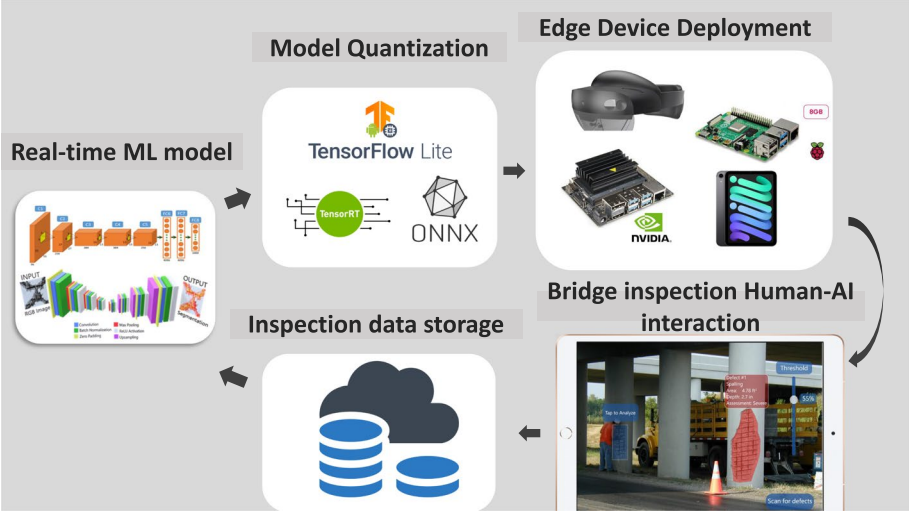

Fig. 1 Illustration of Advanced bridge inspection using a real-time machine learning-based computer vision algorithm on an edge device

study provides a baseline for future semi-automated inspection. Figure 1 demonstrates the proposed methodology.

1.1 Current practice and research

Efforts to use computer vision technology to detect visual defects on the surface of a structure have been going on for almost a decade. Early approaches included edge detection, template matching, and segmentation (Abdel-Qader false, 2003). For detection of spalling and cracks, German et al. (2012) used an entropy-based thresholding algorithm in conjunction with image processing methods in template matching and morphological operations (Abdel-Qader false, 2003). Nguyen et al. (2014) also generated a filter-based algorithm for edge detection, to extract crack edges on the surface of concrete (Nguyen false, 2014). Recent advances in AI and machine learning-based computer vision algorithms have revolutionized many computer vision tasks including the detection and measurement of structural defects.

To process the image data of concrete defects, researchers in the literature implemented the Convolutional Neural Network (CNN) to perform automatic crack detection on concrete surfaces. Combined with transfer learning and data augmentation, Yokoyama and Matsumoto (2017) developed a CNN-based crack detector with 2000 training images (Yokoyama & Matsumoto, 2017). Similarly, Zhang et al. (2020) used an UNet-based algorithm to detect cracks pixel-wise in a more precise manner (Zhang false, 2020a). Zhang et al. (2020) used a CNN-LSTM model to detect, in real-time, concrete bridge deck cracks on a frequency domain (Zhang false, 2020b). The algorithm extracts the crack from its background, where conventional edge detection-based approaches only segment the crack edges; thus the algorithm offers a more feasible solution for crack thickness identification. Deng et al. (2020) uses faster Region-based CNN architecture to automatically detect concrete cracks even in the presence of handwriting (Deng false, 2020). Adhikari et al. (2014) used 3D visualization of crack density by projecting digital

images and neural network models to predict crack depth and the necessary information for condition assessment of concrete components (Adhikari false, 2014). A study by Ren et al. (2020) introduced a CNN model named CrackSegNet to conduct a dense pixel-wise segmentation of cracks in tunnels. The authors used 409 images, collected from a single tunnel, as the training dataset (Ren false, 2020). Multiple studies have also been conducted on measuring and classifying of pavement cracks. One recent study by Eslami and Yun designed an attention-based CNN network which improved the classification results of pavement cracks when compared to automated systems (Eslami & Yun, 2021). While much of the available research focus on detection and measurement of cracks on concrete surfaces or pavements, more recent studies also focus on the detection and quantification of other surface defects. A study by Xu et al., uses Mask R-CNN for automatic defect detection in tunnels. The authors endow a path augmentation and edge detection branch to the network to improve its accuracy, however, this decreased the computational efficiency significantly. The authors focus on detecting leakage and spalling as well as other objects in tunnels such as pipes and edges. Another study by Zhang et al. uses simultaneous object detection and segmentation to segment cracks, spalling, and exposed rebar on the surface of the concrete. The model consists of two separate parts, Yolov3 and Sub Mask architecture, where the bounding box task is conducted separately from the segmentation. The study uses a total of 1440 images for training and testing. The proposed method shows a relatively good speed; however, it is not fast enough for real-time inspection.

A review paper on computer vision-based detection and condition assessment of concrete infrastructure defects emphasizes the importance of sufficiently large, publicly available, and standardized detests -similar methods of imaging and camera specs- to leverage the power of existing supervised machine learning methods for damage detection methods (Koch false, 2015). There are limited available open-source datasets for concrete defects. These includeSDNET2018 (Dorafshan false, 2018) with 58,000 cropped images of cracks on concrete surfaces; CODEBRIM with approximately 1700 original cropped and uncropped images of surface defects, including cracks, spalling, efflorescence and rusting (Mundt & Majumder, 2019); and finally COCO-Bridge which includes 774 images with 2500 instances of bridge elements (Bianchi false, 2021). While the available datasets are adequate for classification purposes, the number of open-source original images needs improvement for automated defect localization and quantification.

This study proposes a methodology to overcome the challenges in data scarcity by using a real-time interactive platform.

2 Real-time machine learning system

The machine learning system introduced in this study is composed of a multi-step process for localizing and quantifying surface defects. First, the defect localization model locates the surface defects in the view of the inspector, creating an attention region for the subsequent defect quantification model. After the localized defects are verified by the inspector, the defects are quantified and measured for condition assessment (e.g., maximum crack width or area of spalling). For defect localization, a deep convolutional neural network (CNN) was trained. CNNs are widely used in object detection tasks including autonomous driving to locate road objects

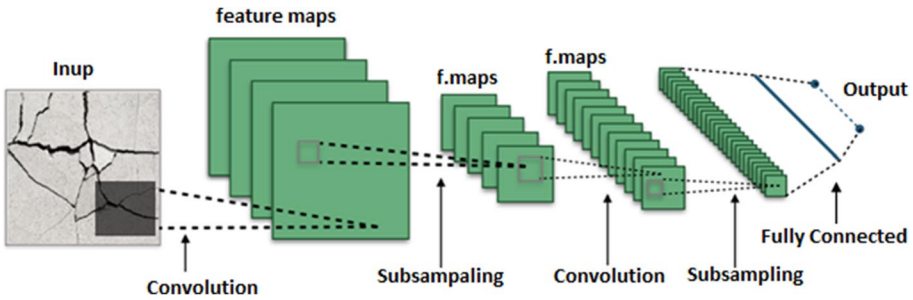

Fig. 2 Architectural diagram of a typical convolutional neural network

(Karaaslan false, 2020), as well as with medical imaging for early detection of abnormalities (Mansoor false, 2015). For quantifying surface defects, another type of CNN model widely used for semantic segmentation tasks was trained. Similarly, semantic segmentation models have shown reliable results in other fields, such as in identifying salient elements in medical scans (LaLonde & Bagci, 2018) or finding lane lines in autonomous driving (Karaaslan false, 2020). CNNs have also already shown promising results in classification and detection of concrete surface defects (Karaaslan false, 2021a; Karaaslan false, 2022a; Zhang false, 2021).

CNNs work well with two-dimensional data and therefore are a great tool to analyze images and videos. These models are typically composed of convolution, pooling, and activation layers to extract features, reduce dimensions for efficient computation, and introduce nonlinearity (Eslami & Yun, 2021). In the convolutional layers, the input images are multiplied by small distinct feature matrices called kernels, and their summations are normalized by matrix size (i.e., kernel size). By convolving images, similarity scores between every region of the image are assigned to generate the image feature matrix. After convolution, the negative values of similarity in the image matrix are removed in the activation layer. A pooling matrix is used to reduce the size of the resultant matrix. For classification, the resultant matrix is then passed through a fully connected layer to obtain class scores. Finally, the image vectors of the trained images are compared with the input image, and a correspondence score, the highest score value, will indicate the classified label. A typical CNN architecture is shown in Fig. 2.

Training the full network of a CNN model often requires large sets of images (> 50,000 images) to obtain the desired accuracy. Due to the difficulty of collecting real-world data for concrete surface defects and the data scarcity, transfer learning is implemented during model training. Transfer learning is a machine learning approach where a previously developed model for a different task is reused as a starting point.

Hyper-parameters also play an important role in the learning rate and accuracy of the model. There are different methodologies on how to select these parameters. For most CNN architectures, the initial learning rate, learning rate decay, batch size, number of epochs, and momentum are the most common hyper-parameters. More complex architectures may have as many as 25 hyper-parameters, making the optimal parameter search a more difficult task. Traditional methods of hyper-parameter searches, such as grid searches, are computationally expensive due to the high dimension of the search

and the unknown correlation among dimensions. In this study, the authors initially used the pre-determined hyper-parameter values from transfer learning and implemented hyper-parameter evolution which further optimized the hyper-parameters using Genetic Algorithm.

When evaluating machine learning models, classifying the predictions as true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN) is a common practice. However, evaluating object detection models will require additional metrics to measure the accuracy of the detected bounding box coordinates. Mean Average Precision (mAP) is a performance indicator that finds the average of maximum precisions at different recall values based on a confidence threshold. Average Precision (AP) is calculated as follows:

$$Presicion = \frac{TP}{TP + FP} \tag{1}$$

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

$$AP = 2 \times \frac{presicion \times recall}{precision + recall} \tag{3}$$

To calculate the precision and recall, TP, FP, and FN need to be determined from evaluation metrics. One of the common metrics is the Intersection over Union (IoU). IoU is also known as the Jaccard Coefficient as it was first introduced by Jaccard (1912). This metric is simply the ratio between the intersection and the union of the predicted boxes, and the ground truth boxes.

$$IoU = \frac{area\ of\ overlap}{area\ of\ union} = \frac{TP}{TP + FP + FN} \tag{4}$$

2.1 Localizing concrete surface defects

A well-performing object detection model typically requires a large dataset with diverse background features and as many object instances as possible for optimal performance. An unbalanced, uniform dataset would lead to the model overfitting the training data and reducing real-world performance for different structure types, materials, or environments. In this study, crack and spalling images were gathered from real-world field data collected at multiple concrete bridges in Florida and also from datasets published by other researchers including CODEBRM and SDNET2018 (Dorafshan false, 2018). The image dataset for model training and validation were prepared so that they provided enough diversity in the background to represent different environments for the defect localization model. Therefore, a significant portion of the existing images were filtered out during data preparation as they were cropped, too small, or caused uniformity in the dataset. A total of 1600 images, with 1750 instances of crack and 1500 instances of spalling, were gathered. The dataset was then divided into two categories of train (80%) and validation (20%) sets for training. An additional 180 images were later collected

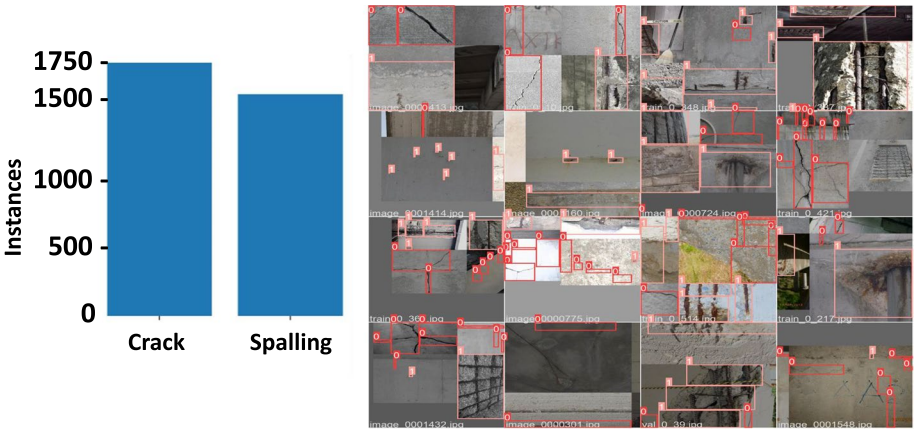

Fig. 3 A sample Training batch and the class distributions for the defect localization model

Fig. 4 Data annotation for the defect localization model

from real-world bridges for testing purposes. Fig. 3 displays the class distribution of a training batch from the dataset.

The data labeling process for defect localization include manually annotating the defect images by drawing bounding boxes around the defects and naming the classes. These boxes will serve as ground truth during training, and each image annotation consists of the defect type and relative pixel location of the box corners. For data labeling, an open-source image annotation tool, LabelMe, was used (Wada, 2016). Figure 4 displays an example of the bounding box annotation of training images.

The authors tested and compared multiple publicly available object detection architectures in order to select the best performing CNN algorithm for defect localization tasks. In order to facilitate real-time processing of the defects, the selected object detection model needed to satisfy two other criteria in addition to high detection accuracy: high

Table 1 Comparison between object detection models

Meta Architecture	mAP (IoU)	Speed (ms)	Speed (FPS)
SSD MobileDet	0.18	0.053	19
SSD MobileNet V3	0.28	0.050	20
YoloV4-tiny	0.37	0.043	23
EfficientNetD0	–	0.250	25
YoloV5s	0.51	0.143	30

inference speed and small memory footprint. Several state-of-the-art CNN architectures, including SSD-MobileNetV3 (Howard false, 2019), SSD-MobileDet (Xiong false, 2020), EfficientDet-D0 (Tan false, 2019), YoloV4-Tiny (Bochkovskiy false, 2020) and YoloV5s (Jocher false, 2021), were tested for mean precision accuracy, inference speed, and quantization capability for edge computing devices.

SSD-MobileNetV3 and SSD-MobileDet have lightweight architecture, suitable for edge computation, and therefore were selected for evaluation in this study. However, none of the models demonstrates the desired accuracy during tests on real-world images collected from a bridge structure. mPA values were also very low and many instances of cracks and spalling remained undetected (Table 1). Yolov3 is another strong CNN architecture that is also popular in autonomous driving (Fan false, 2020). The model has also been used in the localization of defects with good accuracy (Zhang false, 2021). However, the model architecture is large and requires more computational power, making it unsuitable for real-time performance on targeted edge devices in this study. Moreover, the dataset in this study is relatively small for training Yolov3. Yolov4-tiny is another version of the Yolo model that is more suitable for smaller datasets, however the model requires a larger memory size when compared to other CNN architectures, and therefore was not selected for this study. The training results of EfficientNetD0, on the other hand, overfitted constantly. This matter was observed by following the training graph and evaluating the model on test images. Figure 5 displays the loss values from the training and validation sets. The overfitting is observed by comparing the loss function between the training dataset and the validation dataset. The loss value, which represents the summation of errors in a model, usually decreases with each epoch. If the loss decreases only on the training set but remains the same on the validation set or increases, then most likely overfitting has occurred.

Yolov5 is also another powerful object detection model, with high inference speed and a small memory footprint. This object detection model also supports different sizes of datasets. Yolov5S (small) was trained and evaluated in this study. This algorithm also allows for optimization of its hyper-parameters for better accuracy. Initially, using transfer learning, pre-determined hyper-parameters from a previously trained model on the COCO dataset was used. Using a generic optimization algorithm, the hyper-parameters were adjusted for the surface defect dataset. For optimizing mAP values, a weighted combination of metrics was used: mAP@0.5 contributes 10% of the weight and mAP@0.5:0.95 contributes the remaining 90%. After training for 10 epochs, the new optimized hyper-parameters were adjusted to train the defect quantification model. Unlike EfficientNetD0, the loss values in this model dropped consistently after each

Fig. 5 EfficientNeTd0- Loss values per epoch

epoch for both the training and validation sets. The final mAP value reached 0.65. After approximately 400 epochs, the model stopped learning and reached its highest performance. Figure 6 displays the training graphs of Yolov5's object detection. Figure 7 displays its performance on some test images.

2.2 Quantifying concrete surface defects

For concrete defect assessment, it is not enough to detect the damage in a bounding box; the area also needs to be segmented from the intact regions in order to perform quantification, including necessary measurements for understanding the extent of the defects. Therefore, another deep learning model is used sequentially with the defect localization model YoloV5 to perform segmentation of the defect regions. The segmentation model will classify each pixel of the cropped image in the bounding box as "Damage" or "No Damage". Segmentation CNNs in this study consists of an encoder and a decoder section. The encoder is a classification network that conducts the classification, and the decoder semantically projects the features of each class determined by the encoder on the pixel area conducting segmentation. Initially, multiple segmentation models were investigated for their accuracy and inference speed. Popular segmentation models such as FCN (Shelhamer false, 2016), UNet (Ronneberger false, 2015), SegNet (Badrinarayanan false, 2017), and SegCaps (LaLonde & Bagci, 2018) were previously investigated by Karaaslan et al. (2021) (Karaaslan false, 2021b). The study showed that model size (i.e., memory allocation) is the main challenge in deploying these models in edge computing devices. Hence, the authors only evaluated segmentation models with lightweight classifier backbones. Twelve different CNN architectures with different classification backbones were selected for training (4 model architectures with three classifier backbone options). UNet, LinkNet, FPN, and PSPNet segmentation architectures with the backbones of Efficientnetb0, Densenet121, and Inceptionv3 were trained and tested. Figure 8 displays the UNet architecture and and example of its output after the semantic segmentation.

Annotation for the segmentation models is highly sensitive and time consuming. Each pixel on the defect area needs to be annotated separately. Therefore, an open-source annotated dataset was initially used for training. Prior to the training, the dataset was improved using data augmentation. This method artificially increases the dataset by

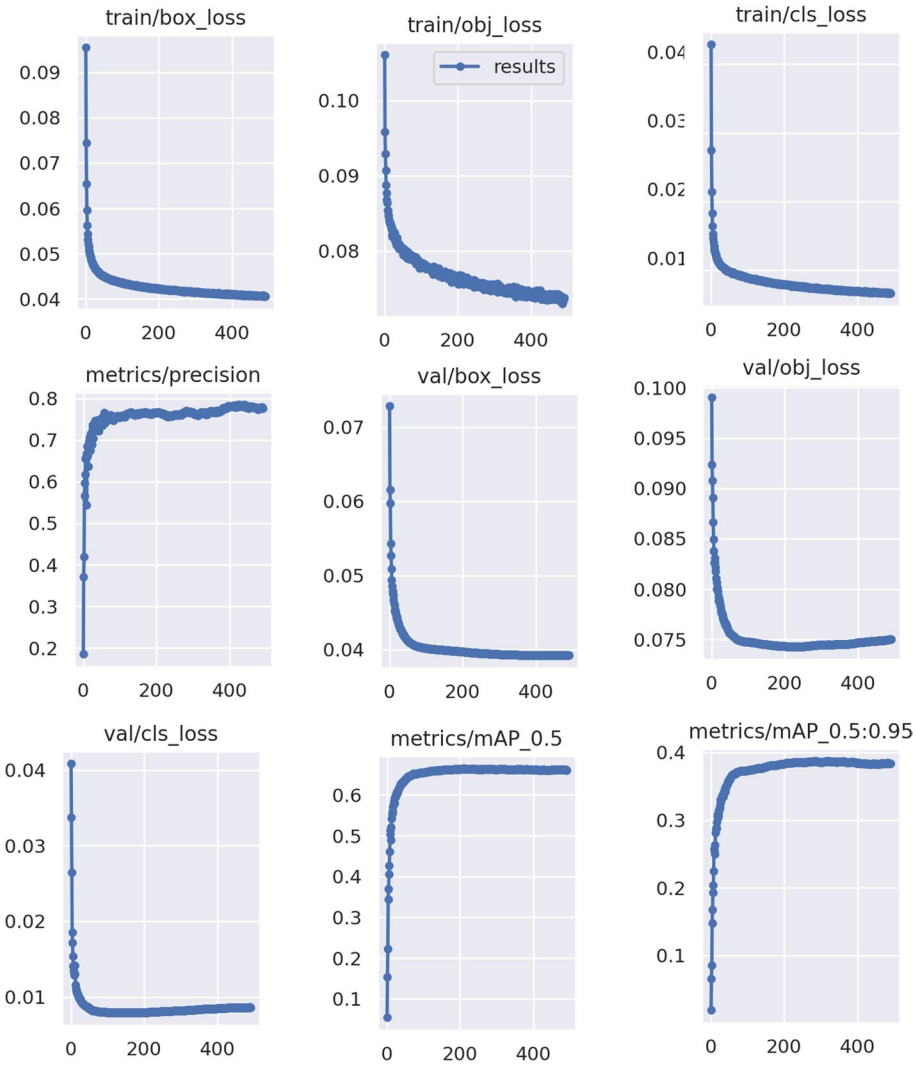

Fig. 6 Yolov5 object detection training metrics

Fig. 7 Yolov5 performance on concrete bridge defects

modifying the images, including cropping, rotation, or adding noise or blur. The images were also resized based on the requirements for each segmentation model. The method improves the accuracy of the model by forming new images for training. Similar to

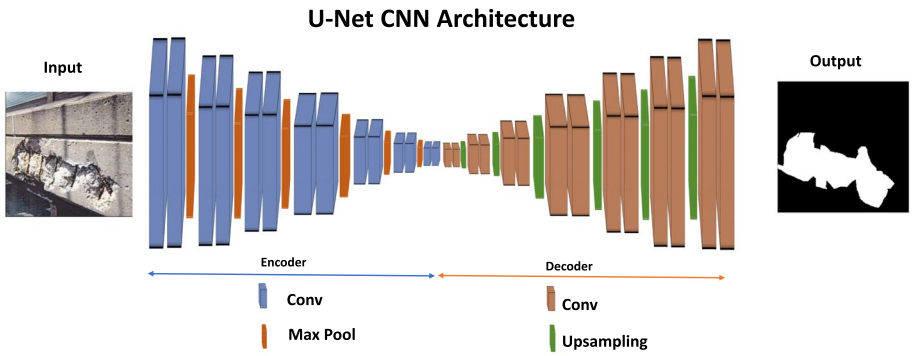

Fig. 8 UNet Architecture for semantic segmentation

defect localization training, transfer learning was also used to train the defect quantification model. In order to improve the training results, a pretrained weight from ImageNet (ImageNet, 2022) was used for the encoder (transfer learning). Initially the model was trained only using ImageNet encoder weight and the decoder from scratch, however, satisfactory results were not obtained in the first run. Therefore, instead of training the decoder from scratch, its layers were frozen for the first couple of epochs and then unfrozen to complete the training. Consequently, the obtained results significantly improved. This process is called fine tuning. Fine tuning is useful when the dataset is small and is used to adjust the already trained weights for the new dataset. Rather than finding the decoder weights from scratch, the pretrained weights are adjusted for the new dataset and used to complete the training.

Table 2 displays the comparison between the models. In the evaluation of the defect quantification model, another metric, F1, which is the weighted average of the precision and recall indicating the model's accuracy, was also used. The model speeds were initially tested on Colab Pro GPU. The training results show that UNet with the backbone of efficientnetb0 is the most accurate model and PSPNet with the same backbone had the highest inference speed. Both criteria are highly important for defect quantification. While inference speed depends on model architecture and cannot be improved, the model accuracy can be improved by adding more data to the dataset. Publicly available annotated image data for segmentation of concrete defects is very limited; therefore a semi-supervised approach was followed to generate more annotated data. First, a small set of the existing data, which already included segmentation labels, was prepared for

Table 2 Comparison between segmentation models

Backbone	Efficientnetb0			Densenet121			Inceptionv3		
CNN Model	IoU score	F1 score	Fps speed	IoU score	F1 score	Fps speed	IoU score	F1 score	Fps speed
UNet	0.74	0.84	16.66	0.55	0.66	16.39	0.41	0.51	15.38
LinkNet	0.56	0.68	12.5	0.45	0.58	11.11	0.58	0.7	14
FPN	0.59	0.71	7.69	0.62	0.74	7.14	0.54	0.67	7.43
PSPNET	0.51	0.63	27	0.56	0.68	26	0.5	0.62	27

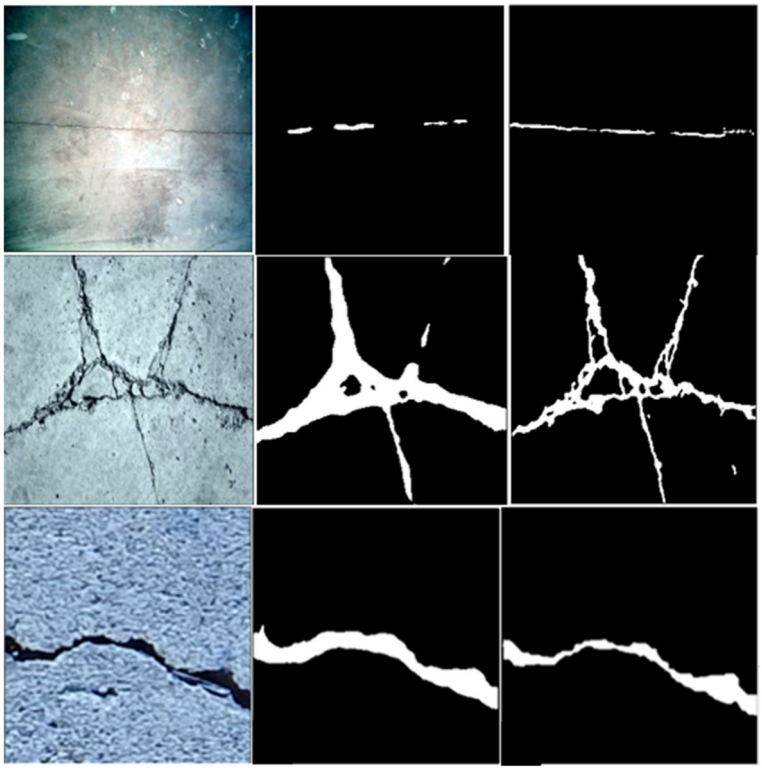

Fig. 9 Semi-supervised annotation improvement

the correct format for the selected training procedure. Then, the segmentation architectures selected for testing were trained for 50 epochs until the models could not improve their precision. The model with the highest accuracy (UNet) was then run over an unlabeled dataset to generate more labeled data. The newly labeled dataset was added to the initial dataset for another round of training. This process was repeated multiple times to obtain accurate annotations. Figure 9 displays the improvement in annotations after 3 rounds of training.

3 Model optimization

Edge devices have limited memory and computational power. Although the developed AI algorithms in this study are lightweight and have a high inference speed, model optimization is still necessary for real-time performance of the models on edge devices. There are three most important properties of the model that need to be reduced for edge computation: storage size, memory usage (RAM), and latency. Model quantization is an optimization method which reduces the size of the model. It can also be used to reduce the amount of time it takes to run an inference through the CNN model (latency). There is a small trade-off between memory size and accuracy. Generally, by reducing the latency and memory size, a small amount of accuracy is lost. Quantization is usually conducted post-training. There are different types of post-training quantization: post-training dynamic range quantization, post-training full integer quantization, and post-training float16 quantization. Some libraries, such as TensorFlow, offer techniques that can be applied on trained weights for quantization. Depending on the processor

on the edge device, the quantization methodologies change. Dynamic range quantization reduces the size of the model to 25% of its original size and improves the model speed by 2–3 times the original value. Dynamic range quantization is suitable for edge devices that run on CPU. Full integer quantization is another method of CNN quantization that decreases the model size by 4 and increases its speed to over 3 times its original value. The integer quantization method is suitable for devices using CPU, edge TPU, and Microcontrollers. Float 16 quantization is also used for CPU and GPUs.

Another important factor in determining the quantization method is the computational power and the required model speed. By using float 16 quantization, the size of floating points is halved; therefore reducing the size of the model by up to half, while causing minimal loss in accuracy. However, it does not reduce the latency of the model as much as quantization to fixed point math. With full integer quantization, more improvement in latency is acquired by quantizing the whole model to integers. As a result, more accuracy is lost. This method needs to convert all variables to integer variables, and therefore, the conversion requires a representative dataset for calibration.

In this study, given the designed ML models, different quantization methods were evaluated for three different edge devices: Raspberry PI with edge-tpu processor, Jetson Nano with NVIDIA Maxwell GPU, and Microsoft HoloLens using Qualcomm Snapdragon 850 CPU. For deployment of the models on Jetson nano with NVIDEA, TensorRT float 16 quantization was used to reduce inference. Float 16 quantization is supported by both Jetson and HoloLens 2. Jetson nano, however, has a more powerful processor, and can handle multiple models performing in real-time with Float 16. Float 16 quantization is not supported by the Raspberry PI with edge-tpu. Instead tflite-edgetpu integer 8 provides satisfactory real-time performance. Some platforms, such as the MR platform in this study, runs in unity and Barracuda and therefore requires an ONNX model. ONNX (Open Neural Network Exchange Format) is a format which can be used for CNN models. The format allows conversions from different libraries including Pytorch and Keras which were used in this study. Float 16 quantization on HoloLens 2 provides a good inference speed, however, by adding more models to the device, more computational power is needed, therefore full integer quantization may be more appropriate.

The final decision on the quantization model depends on the criteria mentioned above and the performance necessary for the task. Each of the mentioned devices as well as many other edge devices, such as tablets and cellphones, can be used for visual inspection. Among these models, MR/AR platforms offer a better human-AI interaction and therefore is recommended. These technologies shape a new environment where physical and virtual objects are integrated at different levels. Due to the development of mobile and embedded devices, together with interactive physical-virtual connections, the customer experience landscape is evolving into new types of hybrid experiences (Karaaslan false, 2022a). In such environments, the physical and digital objects co-exist and interact in real time making it an ideal platform for the inspector to interact with the AI. Using these technologies that are integrated with the proposed method, an inspector can continuously communicate with the AI system. The human-computer interaction in MR will entail practical human-AI collaboration to create a collective intelligence.

4 Defect condition assessment and human-AI interaction

The introduced AI-assisted approach for locating and measuring the surface defects of concrete bridges differs from the available practice in the sense that it provides interaction between the human inspector and the AI (Karaaslan false, 2022b). Instead of a fully automated robotic system that analyzes concrete defects, this study proposes a collaborative method using an AR user interface in which the AI system continuously interacts with the inspector. This interaction eliminates the need for post-processing, reduces labor, and guarantees human-verified results.In order to increase the inspector's involvement in analysis of the defects, two separate AI models are used and optimized to conduct the defect localization and defect quantification in real-time. Defect localization continuously runs on the edge device while the inspector is conducting the routine inspection. Every defect in view is detected and marked using bounding boxes with their class name (crack or spalling). The inspector can communicate with the AI by changing the confidence threshold to ensure the correct detection of all defects. If a bounding box is incorrect or a defect is not detected by the defect localization model, the inspector can modify the AI results, and correct the bounding boxes or draw a new one. The model's threshold, which is the highest probability of detection among the defects and background, is an important factor in the accuracy of prediction and needs to be adjusted based on the environment. For example, an automated system with a threshold of 0.6 would have missed the crack in the scene in Fig. 10, but adjusting the threshold to 0.55 results in accurate detection of all the defects. Upon the approval of the inspector, the AI will run the defect quantification model, which then further analyzes the defect for its size along with determining the defect condition. If the bounding box is not satisfactory, the inspector can adjust the bounding box manually to obtain better results.

Defects in every scene are localized by the defect localization model. Upon the inspector's approval, the detected defect is cropped and used as an input for the defect quantification model. The defect quantification model then provides the size of the defect with respect to its pixel coordinates. Accurate retrieval of real-world dimensional properties from the AR projection is necessary to estimate the condition of the concrete surface defects. Depending on the edge computing device, camera, and the platform that is used (MR, AR), the transformation methods may vary. Mixed reality headsets have dimension

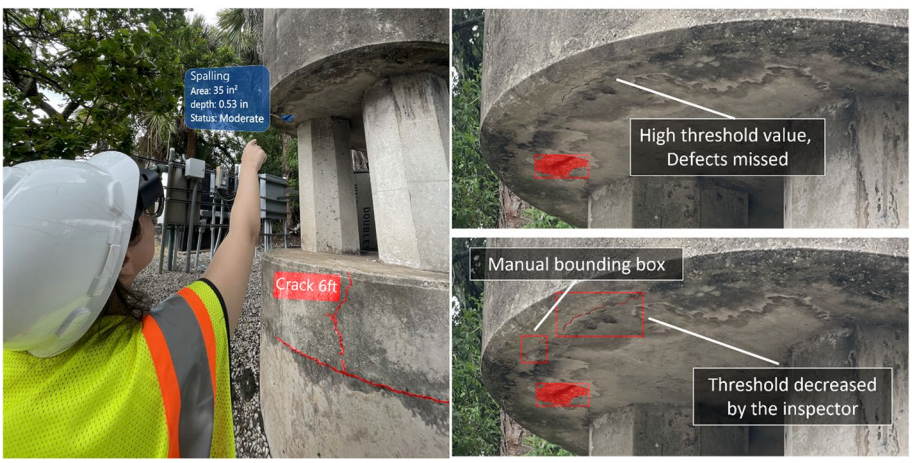

Fig. 10 Human-AI interaction using the MR platform and threshold and bounding box adjustments

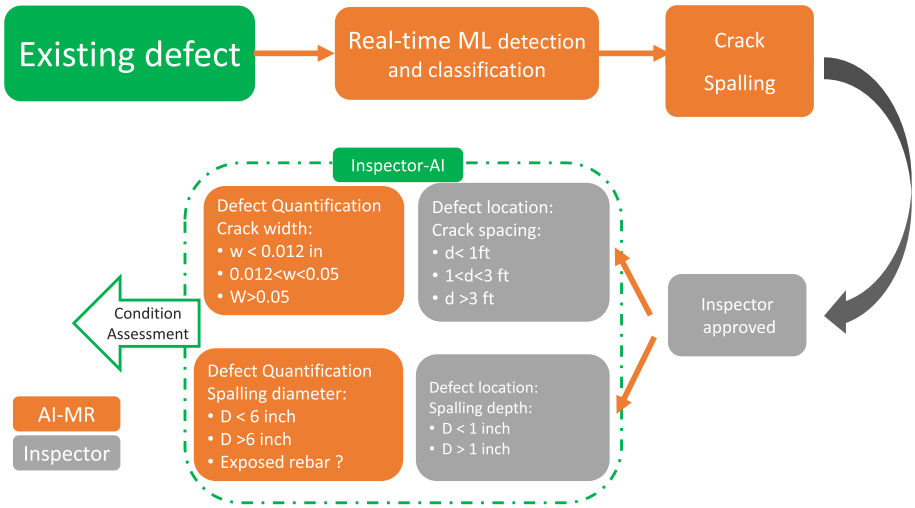

Fig. 11 Defect condition assessment in the proposed methodology with human-AI collaboration

packages and sensors which can conduct the transformation between the pixel coordinates to real-dimensional coordinates. However, the core transformation principle on every edge device is the same. Homography matrixes are one of the most common methodologies to convert dimensions from pixel values to real world values. To assess the severity of a defect during an inspection, the inspector has access to the design details of the bridge, therefore, the inspector can input defect location manually after each analysis of a defect. In general, crack severity is either measured based on its width or its density (Washer false, 2019). Crack width in this study is measured using a defect quantification model and a conversion algorithm obtained to transform the dimensions from pixel coordinates to global coordinates. The area of the spall is the second variable that is being calculated using the defect quantification model and transformation algorithm. However, the area of the spalling is not the only factor in assessing the condition of the defect. Spalling depth and the amount of exposed rebar are also important in assessment of spalling. To tackle this issue, the Human-AI interface allows the inspector to input the other important factors for condition assessment. Figure 11 demonstrates the algorithm for defect assessment.

5 Conclusions

Various studies in the past couple of years have employed learning-based methods for the detection of concrete defects while replacing human involvement. The main purpose of this study, however, is to combine engineering expertise with the efficiency of real-time machine learning using edge computation. The sequential approach of defect localization and defect quantification allows the inspector to verify the results at each step. In addition to that, the segmentation model only runs on the defected area which improves its accuracy and results in better calculation of the defect area. An AR/MR environment is the ideal environment for the collaboration between humans and AI. It enables the human-centered AI to interact with the inspector instead of completely replacing human involvement during inspections. The designed models were tested in different available edge computing devices for performance assessment. It was seen that

with the right quantization method, the algorithm is capable of conducting real-time analysis. Depending on the device and the camera that is being used, camera calibration may be necessary to obtain the physical dimensions of the defects.

Another advantage of this methodology when compared to the current practice is that it generates ready to use annotated data after each inspection. The inspection results can be saved on local hard drives or a cloud, and the data can be eventually used for finetuning the model. This process improves AI performance significantly after each inspection and eventually lead to a reliable model for automated inspection.

One limitation in this study is that the segmentation results are only projected onto planar surfaces since the created image targets are two-dimensional. Therefore, volumetric calculations from curved surfaces (e.g., circular columns) have intrinsically large error. Some of the shortcomings were tackled by using human involvement. Depending on the edge device that is being used, the inspector can either get an approximation of the depth using the AR/MR platform, or by simply using his or her judgment.

Advanced bridge inspection methodology using real-time machine learning can be expanded in many ways in future works on this study. This current work aimed for a generic approach for infrastructure inspections, but the target defect types were only spalls and cracks. In future works, other defect types will be investigated including exposed rebar, efflorescence, and sub-concrete defect (via multi-channel input with infrared data). Moreover, a prototype is being developed using a MR platform for real-time bridge inspection on the edge device.

Abbreviations

AP	Average Precision
AR	Augmented Reality
FN	False Negative
FP	False Positive
FPS	Frame per Second
IoU	Intersection over Union
mAP	mean Average Precision
MR	Mixed Reality
SHM	Structural Health Monitoring
TP	True Positive

Acknowledgements

The authors would like to thank Prof Ulas Bagci of Northwestern University (formerly with University of Central Florida/UCF) and Prof Joe Laviola II of UCF for their collaboration and input. The authors would also like to thank members of UCF CITRS (Civil Infrastructure Technologies for Resilience and Safety) for their help throughout different stages of the project. The project presented in this paper was partially funded by NCHRP IDEA Project 222 (IDEA Program Transportation Research Board of The National Academies). The authors are grateful for their support. The findings and results presented in this paper are the views of the authors, not necessarily the views of the sponsoring agencies.

Authors' contributions

MZ collected and analyzed data, and prepared the manuscript. EK supported data analysis and reviewed the document. FNC planned overall research design, raised funding, and reviewed the manuscript. All authors read and approved the final manuscript.

Authors' information

Ms. Mahta Zakaria is a doctoral student at the Civil Infrastructure Technologies for Resilience and Safety (CITRS) Lab of University of Central Florida (UCF). Her research interests include structural health monitoring using dynamic methods, computer vision, machine learning, and artificial intelligence. She has industry experience in implementing advanced data analytics to a variety of machine vision problems. She also teaches as an adjunct professor at Florida Polytechnic University.

Dr. Enes Karaaslan completed both his doctoral studies and his post-doctoral studies at the (CITRS) Lab of UCF. He is currently the CEO of the start-up company ConnectedWise LLC at the UCF Business Incubator, where he focuses on advanced technologies such as connected autonomous vehicles.

Dr. F. Necati Catbas is an educator and researcher currently serving as a Lockheed Martin St. Laurent Professor at the University of Central Florida. Dr. Catbas is the founding director of Civil Infrastructure Technologies for Resilience and Safety (CITRS) (https://www.cece.ucf.edu/CITRS/). Dr. Catbas and his team focus on theoretical, experimental, and applied

aspects of structural identification, structural health monitoring, nondestructive evaluation, and condition assessment of structural systems. Dr. Catbas is an oelected Fellow of the American Society of Civil Engineers (ASCE) and the Structural Engineering Institute (SEI).

Funding
This project was funded by NCHRP-IDEA program 222.

Availability of data and materials
The datasets used and/or analyzed during the current study are available from the corresponding authors upon reasonable request.

Declarations

Competing interests
F. Necati Catbas is an editorial board member for Advances in Bridge Engineering and was not involved in the editorial review, or the decision to publish, this article. All authors declare that there are no competing interests.

Received: 30 August 2022 Accepted: 4 November 2022
Published online: 28 December 2022

References

Abdel-Qader I, Abudayyeh O, Kelly ME (2003) Analysis of edge-detection techniques for crack identification in bridges. J Comput Civ Eng 17:255–263. https://doi.org/10.1061/(asce)0887-3801(2003)17:4(255)
Adhikari RS, Moselhi O, Bagchi A (2014) Image-based retrieval of concrete crack properties for bridge inspection. Autom Constr 39:180–194. https://doi.org/10.1016/j.autcon.2013.06.011
Badrinarayanan V, Kendall A, Cipolla R (2017) SegNet: a deep convolutional encoder-decoder architecture for image segmentation. IEEE Trans Pattern Anal Mach Intell 39:2481–2495. https://doi.org/10.1109/TPAMI.2016.2644615
Bianchi E, Abbott AL, Tokekar P, Hebdon M (2021) COCO-bridge: structural detail data set for bridge inspections. J Comput Civ Eng 35(3):04021003. https://doi.org/10.1061/(ASCE)CP.1943-5487.0000949
Deng J, Lu Y, Lee VCS (2020) Concrete crack detection with handwriting script interferences using faster region-based convolutional neural network. Comput Civ Infrastruct Eng 35:373–388. https://doi.org/10.1111/mice.12497
Dong CZ, Catbas FN (2021) A review of computer vision–based structural health monitoring at local and global levels. Struct Heal Monit 20:692–743. https://doi.org/10.1177/1475921720935585
Dorafshan S, Thomas RJ, Maguire M (2018) SDNET2018: an annotated image dataset for non-contact concrete crack detection using deep convolutional neural networks. Data Br 21:1664–1668. https://doi.org/10.1016/J.DIB.2018.11.015
Eslami E, Yun HB (2021) Attention-Based Multi-Scale Convolutional Neural Network (A+MCNN) for Multi-Class Classification in Road Images. Sensors 21(15):5137. https://doi.org/10.3390/S21155137
Karaaslan E, Bagci U, Catbas FN (2021a) Attention-guided analysis of infrastructure damage with semi-supervised deep learning. Autom Constr 125:103634. https://doi.org/10.1016/j.autcon.2021.103634
Karaaslan E, Necati CF, Bagci U (2021b) A novel decision support system for Long term management of bridge networks. Appl Sci, 11, 13, 5928
Karaaslan E, Zakaria M, FNC (2022b) In: Alavi A, Feng M, Jiao P, Zahra Sharif-Khodaei B-H (eds) The Rise of Smart Cities: Advanced Structural Sensing and Monitoring Systems, 1st edn. Butterworth-Heinemann
Koch C, Georgieva K, Kasireddy V et al (2015) A review on computer vision based defect detection and condition assessment of concrete and asphalt civil infrastructure. Adv Eng Informatics 29:196–210. https://doi.org/10.1016/j.aei.2015.01.008
Mansoor A, Bagci U, Foster B et al (2015) Segmentation and image analysis of abnormal lungs at CT: current approaches, challenges, and future trends. Radiographics 35:1056. https://doi.org/10.1148/RG.2015140232
Mundt M, Majumder S (2019) Murali S, et al. COncrete DEfect BRidge IMage Dataset, CODEBRIM. https://doi.org/10.5281/ZENODO.2620293
Nguyen HN, Kam TY, Cheng PY (2014) An automatic approach for accurate edge detection of concrete crack utilizing 2D geometric features of crack. J Signal Process Syst 77:221–240. https://doi.org/10.1007/s11265-013-0813-8
Ren Y, Huang J, Hong Z et al (2020) Image-based concrete crack detection in tunnels using deep fully convolutional networks. Constr Build Mater 234:117367. https://doi.org/10.1016/j.conbuildmat.2019.117367
Yokoyama S, Matsumoto T (2017) Development of an automatic detector of cracks in concrete using machine learning. Procedia Eng 171:1250–1255. https://doi.org/10.1016/j.proeng.2017.01.418
Zhang L, Shen J, Zhu B (2020a) A research on an improved Unet-based concrete crack detection algorithm. Struct Heal Monit 20(4):1864–1879. https://doi.org/10.1177/1475921720940068
Zhang Q, Barri K, Babanajad SK, Alavi AH (2020b) Real-time detection of cracks on concrete bridge decks using deep learning in the frequency domain. Engineering. 7(12):1786–1796. https://doi.org/10.1016/j.eng.2020.07.026
Zhang C, Chang CC, Jamshidi M (2021) Simultaneous pixel-level concrete defect detection and grouping using a fully convolutional model. Struct Heal Monit 20:2199–2215. https://doi.org/10.1177/1475921720985437
ARTBA Bridge Report. https://artbabridgereport.org/. Accessed 28 Jul 2022
Bochkovskiy A, Wang C-Y, Liao H-YM (2020) YOLOv4: optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934
Fan J, Huo T, Li X (2020) A review of one-stage detection algorithms in autonomous driving. 2020 4th CAA Int Conf Veh control Intell CVCI 2020 210–214. https://doi.org/10.1109/CVCI51460.2020.9338663

Howard A, Sandler M, Chu G, et al (2019) Searching for MobileNetV3. In Proceedings of the IEEE/CVF international conference on computer vision. p. 1314–1324.
ImageNet. https://www.image-net.org/update-mar-11-2021.php. Accessed 31 Jul 2022
Jocher G, Stoken A, Borovec J et al (2021) ultralytics/yolov5: v5.0 - YOLOv5-P6 1280 models, AWS, Supervisely and YouTube integrations. Zenodo 11. https://doi.org/10.5281/ZENODO.4679653
Karaaslan E, Zakaria M, Ercan T et al (2020) Encrypted transfer of traffic sign information for advanced driving assistance systems using invisible security patches. Transp Res Rec. p. 03611981221119465
Karaaslan E, Zakaria M, Catbas FN (2022a) Mixed reality-assisted smart bridge inspection for future smart cities. Rise Smart Cities:261–280. https://doi.org/10.1016/B978-0-12-817784-6.00002-3
Lalonde R, Bagci U (2018) Capsules for object segmentation. arXiv preprint arXiv:1804.04241.
Ronneberger O, Fischer P, Brox T (2015) U-net: convolutional networks for biomedical image segmentation. In International Conference on Medical image computing and computer-assisted intervention. Springer, Cham, p. 234–241
Seo J, Wacker JP, Duque L (2018) Evaluating the use of drones for timber bridge inspection. Gen Tech Rep - For Prod Lab USDA For Serv 1. FPL-GTR-258. Madison, WI: US Department of Agriculture, Forest Service, Forest Products Laboratory. 1-152., 258, p.1–152.
Shelhamer E, Long J, Darrell T (2016) Fully convolutional networks for semantic segmentation. IEEE Trans Pattern Anal Mach Intell 39(4):640–651. https://doi.org/10.1109/TPAMI.2016.2572683.
Tan M, Pang R, Le QV (2019) EfficientDet: scalable and efficient object detection. Proc IEEE Comput Soc Conf Comput Vis Pattern Recognit:10778–10787. https://doi.org/10.1109/CVPR42600.2020.01079
Wada K (2016) Labelme: Image Polygonal Annotation with Python. Git Code.
Washer G, Hammed M, Brown H et al (2019) Guidelines to improve the quality of Element-Level Bridge Inspection Data. *NCHRP Report* 259
Xiong Y, Liu H, Gupta S, et al (2020) MobileDets: Searching for Object Detection Architectures for Mobile Accelerators. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. p. 3825–3834.

Publisher's Note