Journal of Harbin Engineering University
ISSN: 1006-7043

Vol 45 No. 4
April 2024

Object Detection with Detectron2 for Pothole Detection

Joonho Byun[1], Jungjoon Kim[2], Siwoo Byun[2]*

[1] Department of Artificial Intelligence, Sogang University, Seoul, 06584, South Korea
[2] Department of Software, Anyang University, Anyang, 137060, South Korea

Abstract

Damaged road surfaces can cause traffic accidents and negatively affect the tires and suspension of vehicles, potentially contributing to accidents. Potholes are one of the main causes of road damage, and rapid detection is needed to prevent it. This paper proposes a deep learning model based on Detectron2. Detectron is a training inference platform for Pytorch-based object detection and semantic segmentation. In the experiment, a total of 1334 images were trained, resulting in a class accuracy of 0.94 and a mask accuracy of 0.89, indicating good overall performance. The processing time for an image is considered sufficiently applicable in real life, with an average of 0.11 seconds per image. In the experiment, 199 test images were processed in 21.42 seconds, but the processing time varies depending on the number of instances. The proposed model can overcome the limitations of human inspection of thousands of kilometers of roads.

Keywords: Deep Learning; Pothole detection; Road Damage; Detectron2

1. Introduction

Internet of Things (IoT)[1-4] technologies are used to monitor and collect data on road conditions in real time, while (AI) technologies are used to analyze this vast amount of data to predict road conditions and detect defects. Despite ongoing efforts to improve traffic safety, Korea's traffic fatality rate in 2019 was 6.5 deaths per 100,000 people, higher than the OECD average of 5.2[5].

Potholes, which are holes or scratches in the road surface, have recently become a social issue related to traffic safety. Potholes not only cause direct traffic accidents, but also lead to secondary damages such as tire and suspension damage [6,7].

Existing research has focused on detecting road types (urban, rural, highway), road surface conditions (normal, wet, snow, ice), and driving situations (approaching, entering, passing, left turn, right turn, U-turn). In addition, research has been conducted on detection of falling objects, autonomous driving, and cooperative driving.

2. Objectives

2.1 Types and causes of road damage

Road damage that poses a threat to safe driving includes road gradient changes, road flooding, potholes, and road cracks. Each type has different causes and poses different risks. Excessive rainfall or inadequate drainage from sewers can also contribute to their formation, leading to erosion of the road substructure and shortening its life [8,9].

Figure 1: Examples of potholes

Potholes are depressions in the road surface caused by unevenness of the road surface, blistering and water leakage (Figure 1). They come in a variety of sizes and shapes, from small to large, and are mainly caused by

rapidly changing weather conditions, excessive vehicle traffic, and poor road maintenance[10].

2.2 Road damage detection using deep learning

Research in road damage detection using deep learning focuses on the detection and classification of various damages on road surfaces using computer vision and machine learning techniques. The goal is to provide more effective road management and maintenance strategies by processing large amounts of image data and learning damage patterns. This study suggests that deep learning technology can aid in the development of a big data-based road damage management system. This system would allow road managers to monitor road conditions in real time, detect damage early, and take appropriate action.

To design an effective model, a sufficient training dataset is required to enable the deep learning model to identify damage patterns. Image preprocessing, such as resizing, color correction, and denoising, can improve the model's generalization performance. Image enhancement techniques, including horizontal and vertical flipping, rotation, translation, cropping, color conversion, noise addition, distortion, and deformation, should also be considered.

The choice of deep learning model should be based on the type of damage to be detected and the hardware capabilities available. The more complex the model, the more accurate the predictions, but it may be hardware-intensive or require time-consuming processing that is not feasible in real time. A CNN-based model suitable for image processing can be selected, and after adjusting hyper-parameters such as model structure, learning rate, and number of training runs, the performance of the model can be verified through various evaluation metrics [12-14].

2.3 Deep learning using Detectron

Detectron2[15] is a training/inference platform for object detection and instance segmentation. Detectron2 is a model released by FAIR (Facebook Artificial Intelligence Research) and is the latest evolution of the original Detectron1 model. The Detectron1 model was based on Faster R-CNN[16]. A new Mask R-CNN[17] model was released and the Detectron2 model was released by benchmarking it.

Detectron2 is built on top of Pytorch[18] and typically uses a pre-trained network such as ResNet[19], ResNeXt[20], or a Feature Pyramid Network (FPN)[21] as a backbone network. This backbone network can extract features from images and generate feature maps that are useful for object detection and segmentation tasks.

When training with Detectron2, the training process can be abstracted using the engine instead of the training loop we usually implement when building deep learning models, allowing developers to focus on the model development itself.

Detectron2 offers several advantages over the previous Detectron1, including faster speed, higher accuracy, and modularity. The Detectron2 model benchmarked with Mask R-CNN should be better than the Detectron1 model benchmarked with Faster R-CNN (Figure 2). The reason Detectron2 is faster than other open sources is that it is well optimized for Python and has better performance because the computationally intensive parts are implemented in CUDA and C instead of Python.

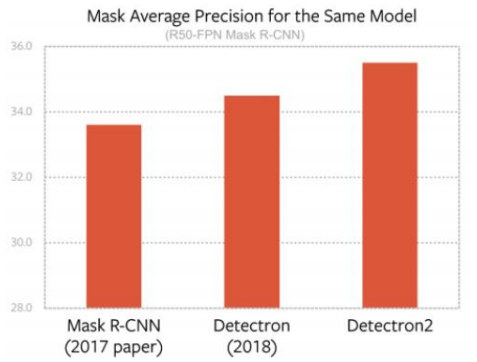

Figure 2: Comparison of Detectron versions

3. Methods

3.1 Pothole dataset

The dataset used in this experiment consists of 1334 images, divided into 875 for the training set, 199 for the validation set, and 260 for the test set (Figure 3). Each image contains at least one instance of a pothole, with a maximum of 10 instances. The data was preprocessed for segmentation purposes using polygon annotation, and the annotation information is stored in a JSON file in COCO-segmentation format[22]. The images were automatically rotated based on the orientation information in the EXIF data, resized to 640 x 640 pixels, converted to grayscale, and no additional data augmentation was applied.

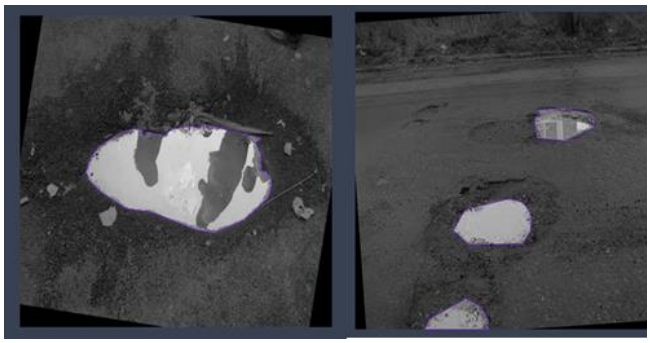

Figure 3: Example image from the pothole dataset.

3.2 Detectron2 Model

The Detectron2 model offers high accuracy by implementing the latest object detection and instance segmentation algorithms. Its modular design allows for easy replacement or addition of different components to meet specific needs. It is based on PyTorch, making it easy to install and use, and comes with pre-trained models that can be easily applied to new tasks. However, training and running models on large datasets requires significant computational resources. Compared to the commonly used Yolo V5 model, Detectron2 generally achieves higher accuracy when the dataset contains less than 150 images. On the other hand, Yolo V5 performs better when the difficulty of classifying or detecting objects is lower. In this experiment, we used the Detectron2 model because the potholes we want to detect have different environments, such as lighting, weather, and road conditions. In addition, the size, type, and number of potholes vary, and in some cases the potholes are cut or covered by shadows, making detection difficult.

3.3 Structure Setups of the Detectron2 Model

3.3.1 Backbone Network

The model's backbone network extracts features from the input image using a combination of FPN and ResNet(Figure 4). FPN generates feature maps of different resolutions, reconstructs low-level features into high-level features, and detects objects of different sizes. Fpn_lateral reconstructs a 256-dimensional feature map using a 1x1 convolution of feature maps from different layers of ResNet. For Fpn_output, a 3x3 convolution is applied to the reconstructed feature maps to produce a 256-dimensional feature map. The Top_block layer performs additional Max Pooling on the highest level feature map. This configuration of the network extracts various levels of features from the input image, enabling object detection and segmentation by considering different sizes and locations of objects.

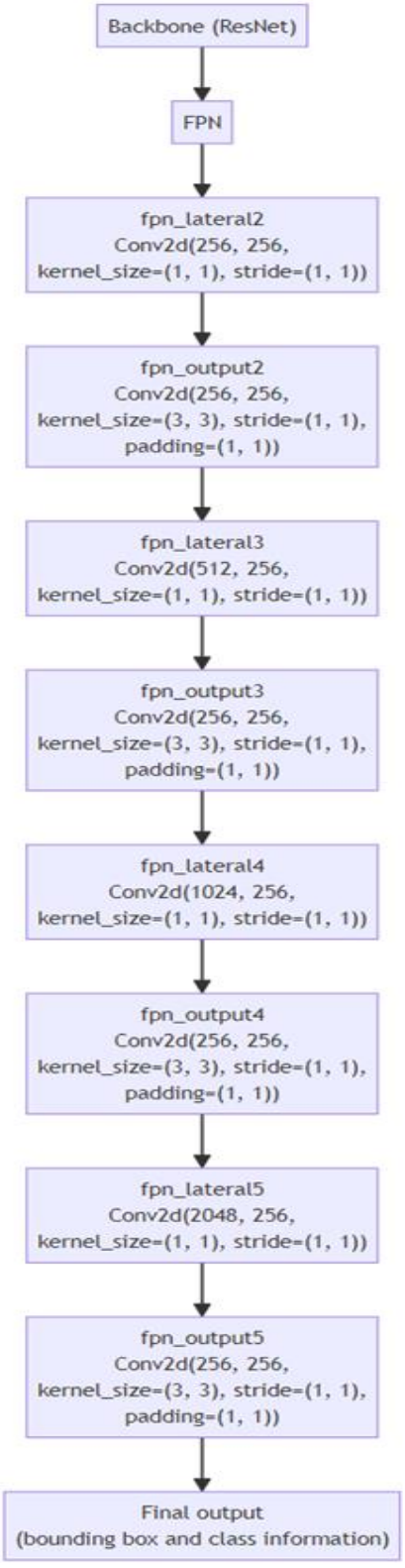

Figure 4: Backbone Network

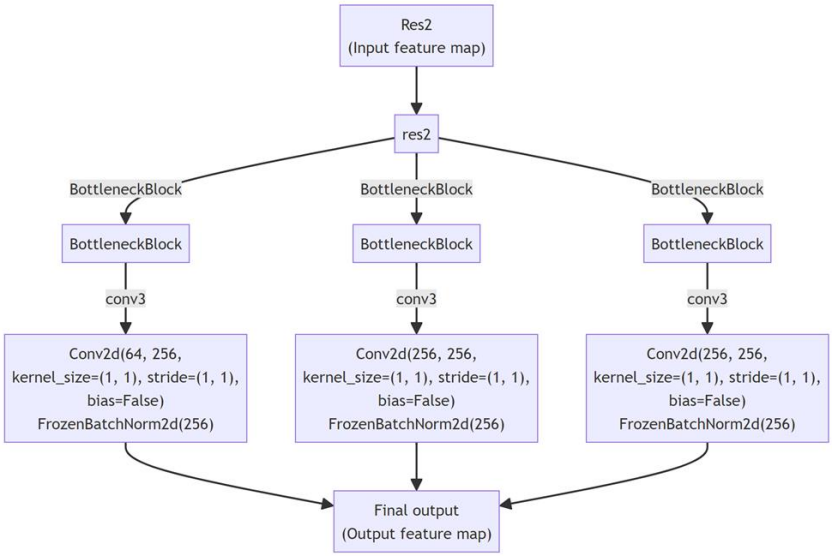

Figure 5: Res2 Block

3.3.2 Res2 Block

ResNet is based on a residual learning mechanism. Each block consists of multiple Bottleneck layers(Figure 5). The block above is the second block, res2. The network is organized in a way that allows for complex representations while minimizing the problem of gradient vanishing. The BottleneckBlock in Res2 includes a shortcut connection layer for training by adding the input directly to the output, a conv1 layer that reduces input dimensionality and model parameters, a conv2 layer that extracts features with a 3x3 operation, and a conv3 layer that increases dimensionality with a 1x1 convolutional operation. The model utilizes res3, res4, and res5 blocks, which have a similar structure. These blocks were not discussed in this paper.

3.3.2 Faster R-CNN Block

The Faster R-CNN structure consists of a Proposal Generator and ROI Heads, which are used to detect and segment objects in an image(Figure 6). The Proposal Generator suggests potential object locations in the image. It takes the output of the backbone network as input and mainly consists of a convolutional layer and a layer for predicting object scores. For ROI Heads, it takes as input the object regions drawn by the Proposal Generator and predicts the location of the object in those regions. The Box Predictor uses the output of the Box Head to predict the class score and bounding box coordinates of the object, while the Mask Head predicts the segmentation mask of the object from the candidate region.

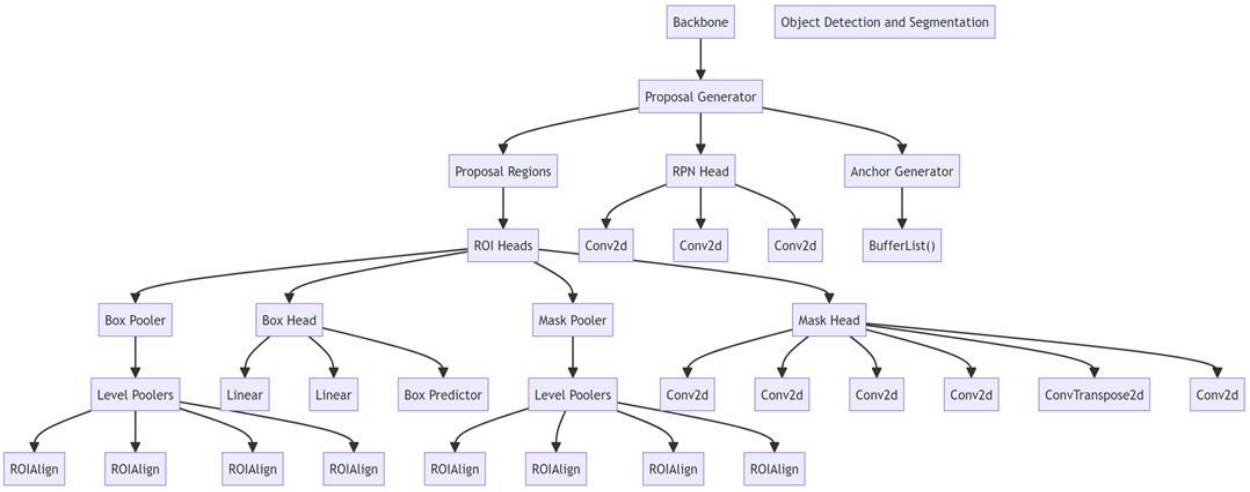

Figure 6: Faster R-CNN Block

4. Results

The model was trained using the weights of the pre-trained Mask R-CNN model on the COCO dataset as initial weights, which were then fine-tuned for the Pothole dataset. The minibatch size, which refers to the number of images used in one training step, was set to 2, and the initial learning rate was set to 0.00025. For the parameter MAX_ITER, we tested 300, 500, 1000, and 2000 candidates and determined that 1000 was the optimal number of training iterations. Therefore, we set it to 1000 iterations. As for SOLVER.STEPS, which decreases as training progresses, we set it to 0 to prevent any decrease.

```
cfg.MODEL.WEIGHTS =
model_zoo.get_checkpoint_url("COCO-
InstanceSegmentation/mask_rcnn_R_50_FPN_3x.yaml")

cfg.SOLVER.IMS_PER_BATCH = 2

cfg.SOLVER.BASE_LR = 0.00025

cfg.SOLVER.MAX_ITER = 1000

cfg.SOLVER.STEPS = []

cfg.MODEL.ROI_HEADS.BATCH_SIZE_PER_IMAGE = 128

cfg.MODEL.ROI_HEADS.NUM_CLASSES = 3
```

In Figure 7, the loss for the class is 0.12 and the accuracy is 0.94, indicating good performance. Similarly, in Figure 8, the loss of the mask that detects the exact boundary of the pothole is 0.25 with an accuracy of 0.89, also indicating good performance. The resulting image shows successful detection of multiple potholes, including small-sized potholes as shown in Figure 9 and multiple potholes as shown in Figure 10. Potholes are often densely packed in a small area, making instance segmentation an effective method for accurately detecting the boundaries of individual objects and separating them from others, as demonstrated in Figure 11. However, some potholes have irregular shapes, such as L-shapes, as shown in Figure 12. To address this issue, we suggest constructing a training dataset that includes a wider range of pothole shapes.

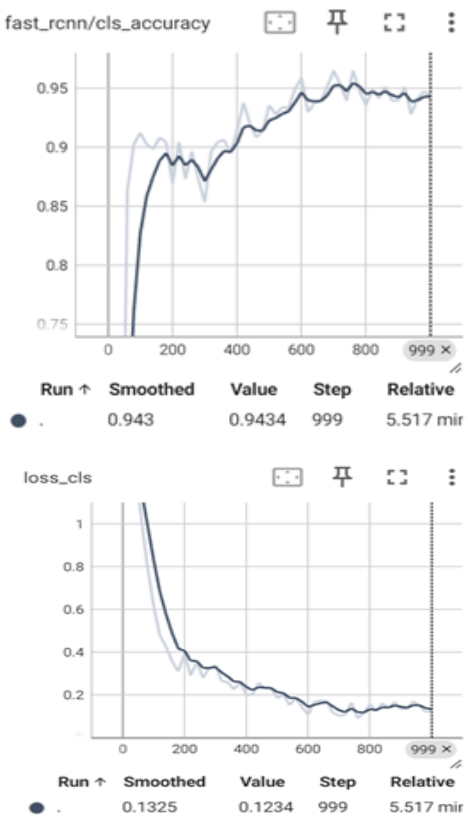

Figure 7: Class_accuracy & loss

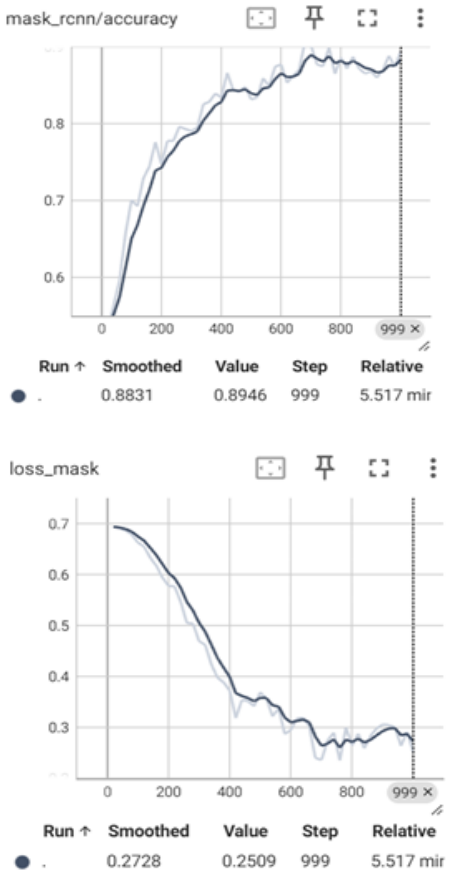

Figure 8: Mask_accuracy & loss

Figure 9: Example of the resulting image 1

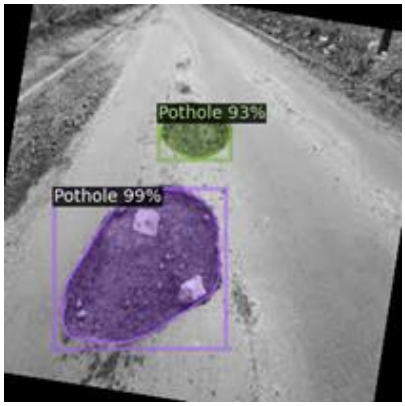

Figure 10: Example of the resulting image 2

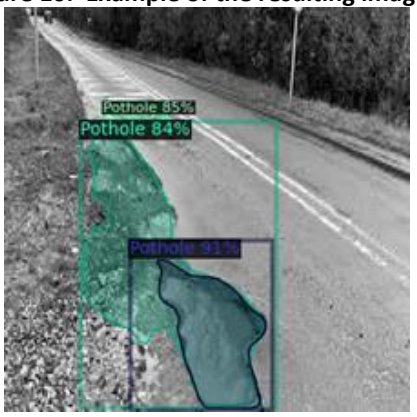

Figure 11: Example of the resulting image 3

Figure 12: Example of the resulting image 4

5. Discussion

In this study, the Detectron2 model was used to detect potholes, one of the main causes of road damage. A total of 1334 images were trained, resulting in a class accuracy of 0.94 and a mask accuracy of 0.89, indicating an overall good performance. The processing time for an image is considered sufficiently applicable in real life, with an average of 0.11 seconds per image. The accuracy increases with the size of the pothole. In the experiment, 199 test images were processed in 21.42 seconds, but the processing time varies depending on the number of instances. Of the potholes that were not detected, most were either too small or obstructed by people, signs, vehicles, shadows, or uneven road conditions. Other objects, such as puddles or wet roads, were also mistaken for potholes. There are several effective solutions to this problem. One is to prepare a high-quality training dataset with a wider variety of potholes. Another is to use data augmentation techniques to improve the generalization performance of the model. In addition, lightweighting the model will enable it to perform well even on low-end hardware in future vehicles. This will help to make pothole detection more widespread and feasible, contributing greatly to safe driving. Improved pothole detection can be achieved by using 3D computer vision or depth estimation technology to estimate the depth of objects. It is also necessary to expand the detection range to include road hazards such as road erosion, road cracks, and road ice in addition to potholes.

Acknowledgment

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No. 2022R1F1A1063659).

Refrences

[1] S. Byun, “Composite Context-based Offloading Scheme for IoT Fault Diagnosis,” Journal of Next-generation Convergence Technology Association, vol. 7, no. 5, pp. 762-773, May 2023. DOI: 10.33097/JNCTA.2023.07.05.762.

[2] L. Ashwini, H. Lee, and M. Hwang, “Implementation of IoT Application using Geofencing Technology for Protecting Crops from Wild Animals,” Asia-pacific Journal of Convergent Research Interchange, vol. 6, no. 6, pp. 13-23, Jun. 2020. DOI: 10.21742/apjcri.2020.06.02.

[3] J. Lee and H. Kim, “Design and Implementation of IoT Devices to Foster Emotions and Prevent Dementia for Senior Citizens,” Journal of Next-

generation Convergence Technology Association, vol. 5, no. 3, pp. 347-353, Jan. 2021.

[4] S. Mansfield, E. Vin, and K. Obraczka, “An IoT-Based system for autonomous, continuous, real-time patient monitoring and its application to pressure injury management,” in Proc. International Conference on Distributed Computing in Sensor System, Pafos, Cyprus, pp. 66-68. 2021. DOI: 10.1109/ DCOSS52077.2021.00024.

[5] J. Kwon, S. Kim, Jae seong Hwang, Jaehyung Lee, and Choul ki Lee, “Development of Prediction Model for Improvement of Safety Facilities in Frequent Traffic Accidents,” The Journal of The Korea Institute of Intelligent Transport Systems, vol.22, no.1, pp.16-24, Dec. 2023. DOI: 10.12815/kits.2023.22.1.16.

[6] G. Ramulu, and B. Murali Krishna, “Analysis of Road Traffic Accidents Due to Potholes and Mitigations using GIS in Hyderabad City,” International Journal of Engineering Research & Technology (IJERT), 2019.

[7] S. kim, K. Kim, and N. Kim, “Suitability Assessment of Asphalt Concrete Pavement Condition Evaluation Index Using Road Crack Density in Small- scale Areas,” Journal of the Korean Society of Civil Engineers, vol. 38, no. 3, pp.467-475, Jun. 2018. DOI: 10.12652/Ksce.2018.38.3.0467.

[8] T. Ha, J. Jung, J. Lee, and S. Lee, “Development of Risk Assessment Scale and Algorithm for Curve Sections,” Journal of the Korean Society of Civil Engineers, vol.28, no. 5D, pp.627-639, Sep. 2008. DOI: 10.12652/Ksce.2008.28.5D.627.

[9] V. Bakhtiari, F. Piadeh, K. Behzadian, and Z. Kapelan, “A critical review for the application of cutting-edge digital visualisation technologies for effective urban flood risk management,” Sustainable Cities and Society, vol. 99, pp. 1-14, 2023. DOI: 10.1016/j.scs.2023.104958.

[10] S.-P. Hong and Y.-J. Cho, “A Study on Road Porthole Detection based on Deep Learning”, Journal of Information Technology and Applied Engineering, vol. 12, no. 2, .pp. 35-41, Aug. 2022. DOI: 10.22733/JITAE.2022.12.02.005.

[11] S. D. Nguyen, T. S. Tran, V.P. Tran, H. J. Lee, Md. J. Piran, and V. P. Le, “Deep Learning-Based Crack Detection: A Survey,” International Journal of Pavement Research and Technology, vol. 16, pp. 943-967, 2Apr. 2022. DOI: 10.1007/s42947-022-00172-z.

[12] J. Chen, P. Tang, T. Rakstad, M. Patrick, and X. Zhou, “Augmenting a deep-learning algorithm with canal inspection knowledge for reliable water leak detection from multispectral satellite images,” Advanced Engineering Informatics, vol. 46, pp. 1-14, Oct. 2020. DOI: 10.1016/j.aei.2020.101161.

[13] E. H. Ali, H. A. Jaber, and N. N. Kadhim, “New algorithm for localization of iris recognition using deep learning neural networks,” Indonesian Journal of Electrical Engi-neering and Computer Science, vol. 29, no. 1, pp. 110-119, Jan. 2023. DOI: 10.11591/ijeecs.v29.i1.pp110-119.

[14] M. Kasian and H. Kilavo, “A comparative study on performance of SVM and CNN in Tan-zania sign language translation using image recognition,” Applied Artificial Intelligence, vol. 36, no. 1, pp. 452-466, Nov. 2021. DOI: 10.1080/08839514.2021.2005297.

[15] R. Girshick, K. He, P. Dollár, and R. B. Girshick, "Detectron2," in Proc. European Conference on Computer Vision (ECCV), 2020.

[16] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," in Proc. IEEE Conference on Computer Vision and Pattern Recognition, 2016.

[17] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in Proc. IEEE International Conference on Computer Vision (ICCV), 2017.

[18] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in PyTorch," in Proc. 31st Conference on Neural Information Processing Systems (NIPS), 2017.

[19] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proc. IEEE conference on computer vision and pattern recognition, 2016.

[20] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated Residual Transformations for Deep Neural Networks," in Proc. IEEE Conference on Computer Vision and Pattern Recognition, 2017.

[21] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Feature Pyramid Networks for Object Detection," in Proc. IEEE Conference on Computer Vision and Pattern Recognition, 2017.

[22] Tsung-Yi Lin, “Microsoft COCO: Common Objects in Context,” in Proc. ECCV, 2014. DOI: 10.1007/978-3-319-10602-1_48