 https://doi.org/10.24191/jeesr.v29i1.006

Comparative Performance Analysis of YOLOv11m and YOLOv11n for an AI-Based Pothole Detection and Reporting System

Aisyah Hannah Mohd Zaki, Norlela Ishak*, Suzi Seroja Sarnin and Mohd Rizal Dohad

***Abstract*—Road potholes remain a significant infrastructure and public safety concern, as conventional reporting systems rely heavily on manual verification and inspection processes that lead to delayed maintenance actions. This paper presents a comparative evaluation of the YOLOv11m and YOLOv11n object detection models for an AI-based pothole detection and reporting system. Both models were trained and evaluated under identical configuration settings using the publicly available Multi-Weather Pothole Detection (MWPD) dataset consisting of 3,087 images captured under diverse environmental conditions. Model performance was assessed using mean Average Precision at IoU 0.5 (mAP50), precision, recall, precision–recall curves, and confusion matrix analysis. Experimental results show that YOLOv11m achieved better performance metrics, recording mAP50 of 0.76692, precision of 0.76728, and recall of 0.7020, compared with 0.74843, 0.75973, and 0.6911 respectively for YOLOv11n. Although the performance differences were relatively small, YOLOv11m demonstrated a better overall balance between pothole localization accuracy and false detections reduction, making it the preferred model for integration into the proposed reporting platform. The developed prototype automatically analyses user-submitted road images, verifies pothole presence, displays detection results, and records reporting information. The results demonstrate the feasibility of integrating deep learning-based object detection into municipal road reporting workflows to improve the efficiency, consistency, and reliability of pothole verification.**

***Index Terms* — Computer Vision, Object Detection, Pothole Detection, YOLOv11m, YOLOv11n.**

I. INTRODUCTION

A pothole is a type of road structural damage characterized by the submerging of road surfaces and is commonly found on paved roads. Paved roads are constructed using mixtures of gravel, aggregate, and asphalt or concrete to provide transportation facilities for vehicle or foot traffic [1]. Asphalt is used on almost 94% of roads today due to its affordability and adaptability to diverse climates and traffic conditions [2]. Despite the durability of asphalt-paved roads, deterioration can still occur, resulting in damaged surfaces that pose risks to road users. According to the Tensar white paper, pothole formation is contributed by factors including surface cracks, water infiltration, and traffic loads. Pavement cracks can allow water to penetrate beneath the surface, weakening the structure and accelerating deterioration under repeated vehicle loads [3]. Potholes consequently pose safety and economic risks, particularly to motorcyclists, while also increasing maintenance costs for authorities and causing vehicle damage such as suspension, bearing, tire, and alignment failures.

To overcome the risks of potholes, road authorities such as Jabatan Kerja Raya (JKR) and Pihak Berkuasa Tempatan (PBT) have implemented road monitoring and maintenance efforts. Kementerian Kerja Raya (KKR) has also developed the MyJalan mobile application to allow civilians to report road damage [4], [5]. However, potholes remain a significant safety concern, with 181 road accidents, including 23 fatal cases, attributed to potholes between 2022 and July 2024 according to the Road Accident Management System (RAMS) and KKR [6]. Furthermore, 17,921 pothole instances were reported on federal roads between January and April 2025, indicating that road deterioration remains a persistent issue [7]. Although reporting platforms such as MyJalan are available, current systems still rely heavily on human input for the identification and verification of road defects. Conventional approaches require personnel to conduct visual inspections or rely on citizen complaints, followed by manual data collection, verification, documentation, and categorization. This dependence on manual processes can introduce human errors and reduce the efficiency of road damage reporting.

Following the rapid expansion of the Fourth Industrial Revolution (IR 4.0), Artificial Intelligence (AI) and automation have been increasingly adopted across various sectors [8], including road maintenance. Advanced computer vision and deep learning technologies enable automated image and video analysis for object detection with reduced human intervention. Local authorities such as Majlis Bandaraya Petaling Jaya (MBPJ) have also integrated AI with urban CCTV systems to automate the detection of public safety and infrastructure-related incidents. However, the effectiveness of such systems is highly dependent on the performance of the AI model used.

A review of existing literature identified several object

This manuscript is submitted on August 28, 2026, revised on October 5, 2026, accepted on October 6, 2026, and published on October 31, 2026.

Aisyah Hannah Mohd Zaki, Norlela Ishak, Suzi Seroja Sarnin and Mohd Rizal Dohad are with the Faculty of Electrical Engineering, Universiti Teknologi MARA (UiTM) Shah Alam, 40450 Selangor, Malaysia

*Corresponding author
Email address: norlelaishak@uitm.edu.my

1985-5389/© 2026 The Authors. Published by UiTM Press. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

detection models with potential for implementation, including the YOLO (You Only Look Once) family. Recent studies have demonstrated that YOLOv11 provides a strong balance between detection accuracy, inference speed, and computational efficiency across various applications [9]–[14]. Beqqali Hassani et al. reported improved overall performance compared to previous YOLO versions, while Alkhammash et al. found that YOLOv11n achieved strong detection performance compared to YOLOv9 and YOLOv10 [10]. Other studies reported advantages in inference speed, precision, accuracy, and mAP across applications including fire detection, bone fracture detection, and mixed-traffic object detection [11]–[13], while its effectiveness has also been demonstrated in oriented bounding box detection [14]. These findings demonstrate the versatility of YOLOv11 for accurate and efficient object detection, motivating this research to compare the YOLOv11n and YOLOv11m variants.

The effectiveness of YOLO architecture has also been demonstrated specifically in pothole detection. Rasee et al. demonstrated that AI-based vision systems can improve the efficiency and consistency of pothole identification compared to manual inspection [15], while Almasri et al. showed that YOLO-based models can provide accurate pothole detection with low inference time [16]. Fortin et al. demonstrated the feasibility of deploying lightweight YOLO models on Raspberry Pi for small-scale, resource-constrained pothole detection [17]. Similarly, Vardhan et al. developed a YOLOv11-based system that extended pothole detection to include dimension, volume, material, labour cost estimation, and automated report generation [18].

The reviewed studies consistently demonstrated that YOLOv11 offers improvements in detection accuracy, inference speed, and computational efficiency, making it a suitable architecture for real-time object detection tasks. The chapter then reviewed previous studies on YOLO-based pothole detection systems, highlighting the feasibility of deep learning for automated road condition assessment and the practical implementation of such systems in intelligent transportation and smart city applications.

II. METHODOLOGY

A. Overall Workflow

Fig. 1 shows the overall workflow flowchart of the project implementation. The overall workflow is divided into three phases, each of which addressed each of the objectives of this project.

B. Phase 1: Dataset Preparation and Model Training

Phase 1 fully covers the development of an AI model for pothole detections, starting from its fundamentals of dataset collection, dataset preparation and finally model training.

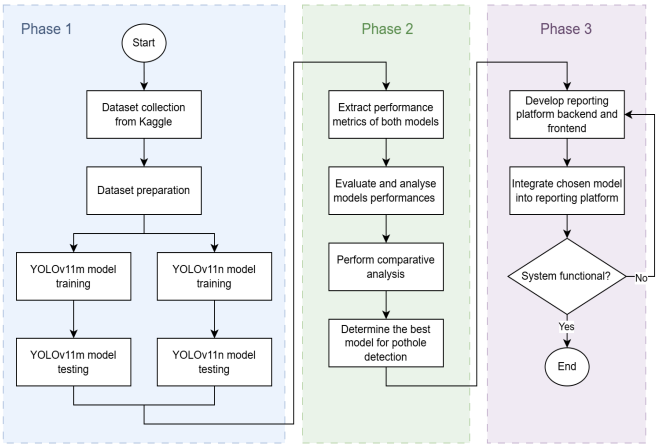

Fig 1. Overall workflow flowchart

1) Retrieving Dataset from Kaggle

To start with the development of pothole detection model, the dataset is first obtained through Kaggle, an open-source data collection platform. The Multi-Weather Pothole Detection (MWPD) dataset is specifically chosen as it provides a collection of pothole images with diverse environmental conditions. Unlike usual dataset images that were captured under ideal conditions, the MWPD dataset contains images taken under various weather, lighting, wet or dry asphalt roads conditions ensuring the development of a robust and adaptable pothole detection model.

The MWPD dataset has gone through several image pre-processing techniques which include image resizing and image augmentation.

a) Image Resizing

Image resizing is a critical data preparation and pre-processing step to standardize the dimensions of all images in the dataset. Certain image detection model such as models with YOLO architecture commonly require input images with the dimensions of 416 × 416 or 640 × 640 dimensions [19]. Standardisation of the dataset dimensions following the architecture requirements will allow the model to process the images efficiently. The image size in the MWPD dataset has a standardized 640 × 640 dimension which is compatible to be applied to YOLO models.

b) Image Augmentation

Another pre-processing technique applied for this dataset is image augmentation. Image augmentation is a data pre-processing technique which uses pre-existing image data to create new data samples and increase the amount and variety of images within the dataset. According to K.Maharana et al. “Data augmentation is described as a strategy to prevent overfitting via regularization” [20]. Image augmentation may include geometric transformations and photometric transformations in which both of them implement image file manipulation techniques.

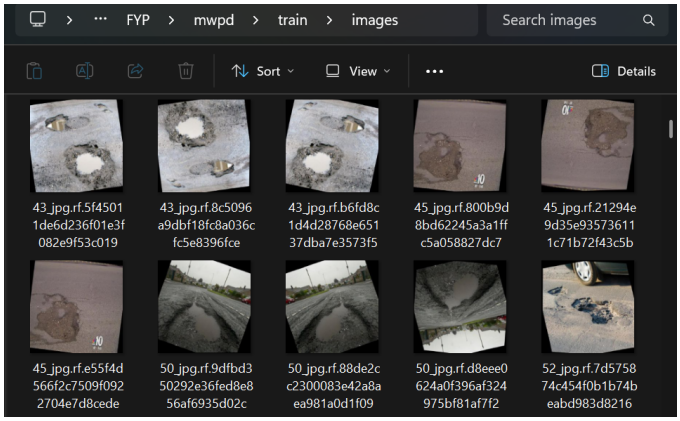

Fig 2. MWPD dataset containing augmented images.

As can be seen in Fig. 2 above, the MWPD dataset is augmented by implementing geometric transformation to tweak the orientation of the original image. These transformations include vertical and horizontal image flipping as well as small-angle image rotations to simulate different camera orientations during image acquisition. No additional photometric transformations, such as brightness adjustment, colour jittering, contrast enhancement or Gaussian noise injection were applied during this project as the dataset had already been pre-processed prior to distribution. By preserving the original augmentation strategy implemented by the dataset publisher, both YOLOv11 models were trained using identical image variations to ensure a fair comparison of their detection performances.

Overall, the MWPD dataset contains a total of 3087 images pre-divided into train, valid, and test folders separating the training, validation, and testing datasets. The dataset is split into a 90:10 ratio where 90% of the dataset is categorized as training dataset, while the other 10% is divided between validation and training dataset. The dataset partition used in this project follows the original folder structure provided by the MWPD dataset, where the images are already separated into training, validation, and testing subsets by the dataset publisher. In addition to that, since the MWPD dataset is relatively small, maintaining a 90:10 training-testing split allows for maximising the data available for model training while retaining a sufficient independent test set for performance evaluation [21]. Therefore, no additional manual reshuffling or redistribution of the images was performed. Maintaining the original dataset partition ensures consistency with the published dataset and allows the model performance to be evaluated under the same experimental conditions intended by the dataset developer.

2) YOLOv11 Class Selection

The YOLOv11 model family consists of five model classes which are Nano (n), Small (s), Medium (m), Large (l), and Extra-Large (x). Although it is possible to evaluate all available classes, this paper focuses on only two models, namely YOLOv11-n and YOLOv11-m.

YOLOv11n is selected due to its designed architecture that is lightweight which is suitable for deployment in resource-constrained devices. Its compact architecture requires fewer computational resources, making it suitable for deployment on edge computing platforms such as embedded systems and single-board computers like Raspberry Pi without relying on cloud processing or high-end Graphics Processing Units (GPUs). Due to its low computational complexity and fast inference speed [22], YOLOv11n is widely adopted in research and practical applications where real-time performance and hardware efficiency are primary considerations [23]

On the other hand, YOLOv11m is selected because it offers a balanced trade-off between detection accuracy and computational requirements. Compared to YOLOv11n, the medium variant has a larger network capacity, enabling it to learn more complex features and potentially achieve higher detection performance while remaining computationally feasible on standard desktop or laptop GPUs. Unlike the larger YOLOv11 variants, YOLOv11m does not require high-end graphics hardware or cloud-based computing, making it a practical choice for general-purpose computer vision applications [24, 25].

3) Model Training

The training of the YOLOv11 models was implemented by using the Ultralytics YOLO Python Library, which allows for a plug and play interface for customized YOLO model development. The Python script was referenced from the usage example provided from the Ultralytics YOLO v11 documentation [26].

The model training was carried out on a local workstation to ensure that both YOLOv11 models were evaluated under identical hardware conditions. The workstation specifications consisted of a Ryzen 5 3600 Central Processor Unit (CPU), GTX 1660 Super GPU with 6 GB DDR6 RAM, 16 GB 3200 MHz RAM, running Windows 11, and Python 3.11. Using the same hardware and software environment ensures that the observed differences in model performance are attributable to the model architecture rather than variations in computational resources.

The training was done in 50 epochs, reduced from default 100 epochs. Preliminary observations during the training process indicated that both YOLOv11 models exhibited stable convergence before reaching 50 epochs, with the training and validation losses showing only marginal improvements in the later training stages. Extending the training beyond 50 epochs was therefore unlikely to provide significant performance improvement while increasing the risk of overfitting. Furthermore, using the same number of epochs for both YOLOv11n and YOLOv11m ensures a fair experimental comparison by maintaining identical training conditions throughout the study. Other training parameters were kept consistent between both models, with a batch size of 16, image size of 640 × 640 pixels, automatic optimizer selection, an initial learning rate of 0.01, and confidence threshold of 0.25.

C. Phase 2: Model Result Output Analysis and Comparative Analysis

Upon completion of the training process, the Ultralytics YOLO library automatically generates a collection of output files containing various performance metrics and

visualizations. These outputs provide quantitative and qualitative insights into the learning behaviour and detection performance of each model [27]. The results obtained from YOLOv11m and YOLOv11n are then analysed and compared to determine which model is more suitable for the proposed AI-based pothole reporting system.

D. Phase 3: Reporting Platform Development and Model Integration

Upon completion of the model comparison, the selected YOLOv11 model is integrated into the mock-up of the pothole reporting platform. A graphical user interface (GUI) is developed using the Python Tkinter library to simulate the functionality of the reporting system. The platform allows users to submit pothole reports by uploading images, which are then analysed by the chosen AI model to verify the presence of a pothole. This phase demonstrates the integration of the trained object detection model into a functional application, illustrating its use in assisting the verification of user-submitted reports before they are processed by the reporting platform administrator.

1) Reporting Platform User Interface

When an image is uploaded through the upload image button of the reporting platform, the integrated AI model will perform object detection to determine whether the image contains a pothole. If a pothole is detected, the system displays the image with the detected pothole along with its detection bounding box and the corresponding confidence score. Conversely, if no pothole is detected, the system informs the user that no pothole is present in the submitted image. Examples of both successful and unsuccessful detections are presented in Fig. 3 below.

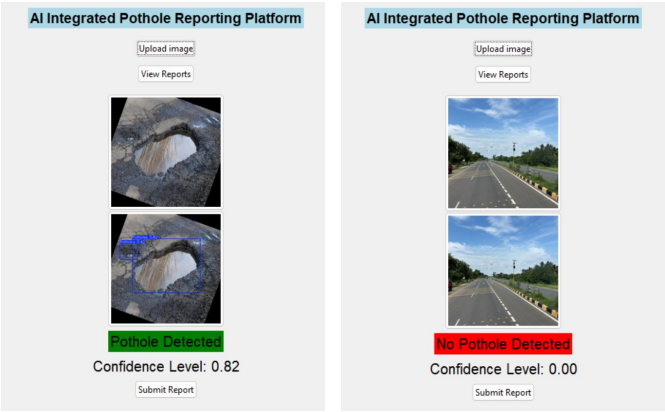

Fig. 3. Output for pothole is detected (left) and no pothole detected (right)

The reporting system mock-up is also equipped with data logging features where past reports along with its detection result can be viewed by clicking the View Reports button and selecting on the specific data log to be recalled. The report data logging UI can be seen in the following Fig. 4.

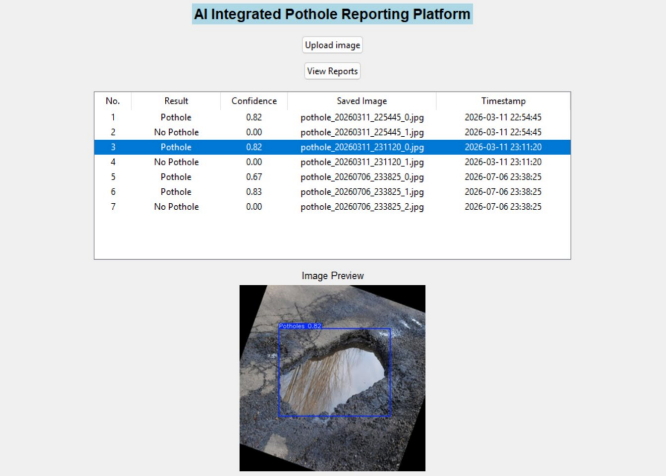

Fig. 4. Historical data of past submitted reports

2) Overall System Architecture

The final system architecture can be summarized into Fig. 5 below.

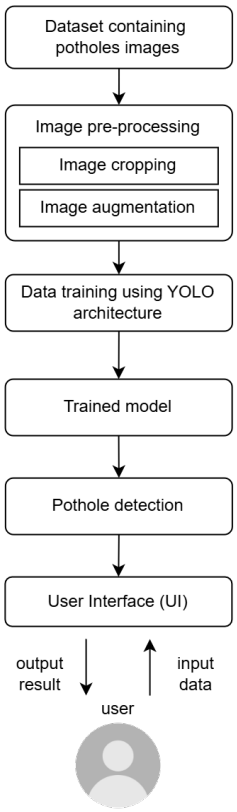

Fig. 5. Final overall system architecture of AI-based pothole detection reporting system

III. Results And Discussion

A. Comparative Analysis

1) Mean Average Precision 50 (mAP50)

Mean average precision (mAP) is a measure on how well the model is at finding objects. mAP50 extends that definition by calculating the mean average precision specifically at an Intersect over Union (IoU) threshold of at least 50%. Which

means that the model's detection needs to cover at least 50% of the real object. This metric is more commonly used when the primary concern of the detection is only to detect the presence of an object without needing to be accurate on its positioning, which is why this metric is significant as the primary metric for the model selection of an AI-based pothole detection reporting platform.

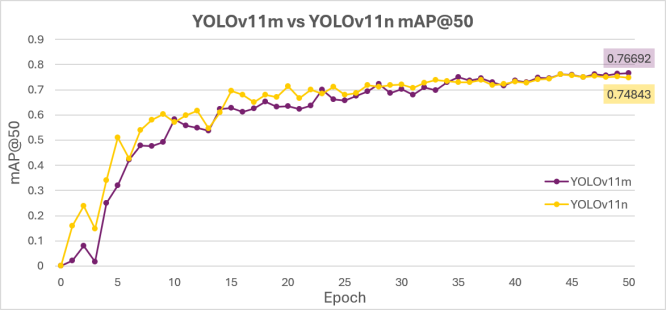

Fig. 6. YOLOv11m and YOLOv11n mAP@50 per epoch graph

2) Recall

Recall is a performance metric that measures the ability of a model to correctly identify all actual positive instances. A higher recall indicates that the model successfully detects a larger proportion of actual potholes while minimizing the number of missed detections (false negatives). Recall is particularly important in the proposed AI-based pothole reporting platform because it reflects the likelihood of the system failing to detect an actual pothole submitted by a user.

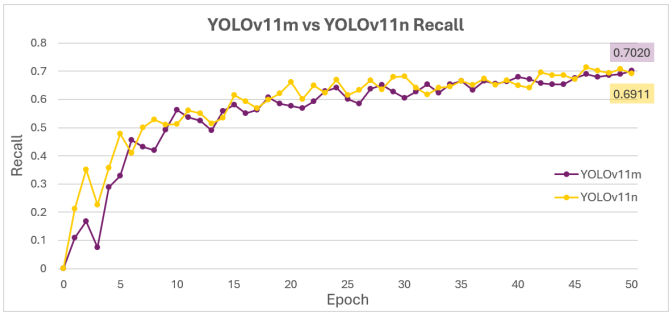

Fig. 7. YOLOv11m and YOLOv11n Recall per epoch graph

3) Precision

Precision is a performance metric that measures the proportion of positive detections made by the model that are actually correct. Higher precision indicates that the model produces fewer false positive detections where non-pothole objects such as road markings, shadows, or surface imperfections, being incorrectly identified as potholes.

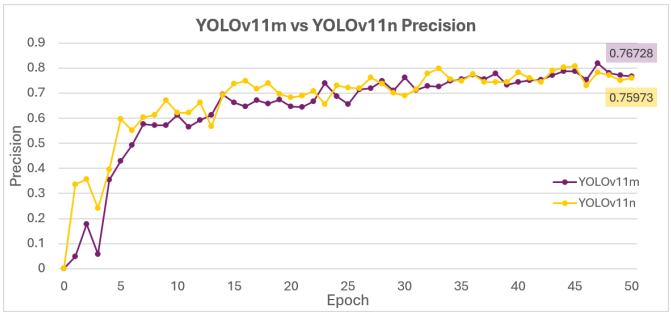

Fig. 8. YOLOv11m and YOLOv11n Precision per epoch graph

TABLE II. MODEL PERFORMANCES SUMMARY

Metric	YOLOv11m	YOLOv11n	Difference
mAP@50	0.76692	0.74843	0.01849
Recall	0.7020	0.6911	0.0109
Precision	0.76728	0.75973	0.00755

Table I shows that YOLOv11m outperforms YOLOv11n model across all metrics.

4) Precision-Recall Curve

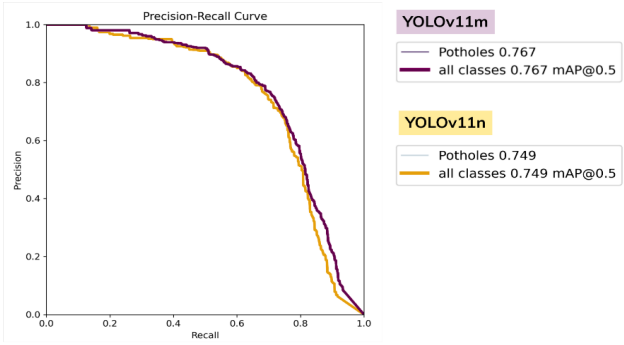

Fig. 9. Precision recall curve YOLOv11m (left) and YOLOv11n (right)

The precision–recall curve analysis shows that YOLOv11m achieved a mAP@0.5 value of 0.767, while YOLOv11n obtained a value of 0.749. This metric represents the area under the precision–recall curve at an IoU threshold of 0.5, which evaluates the model's overall balance between precision and recall across different confidence thresholds. The results indicate that YOLOv11m has a slightly better overall trade-off between correctly detecting potholes and minimizing both false positives and false negatives.

B. Model Selection

The result for each of the performance metrics discussed shows that YOLOv11m outperforms YOLOv11n in terms of detection evaluation metrics by a very small margin. It achieves the results of 0.76692 in mAP50, 0.76728 in precision, and 0.7020 in recall. The difference of 0.018 in mAP50 between them is extremely marginal, indicating that both models achieved comparable detection performance.

Theoretically, YOLOv11m could provide higher detection capability at the cost of increased computational complexity. YOLOv11m has 20.1 million parameters and 68.1 billion FLOPs, compared to 2.6 million parameters and 6.5 billion

FLOPs for YOLOv11n [28]. The reported inference speed on a T4 GPU using TensorRT10 is 4.7 ms for YOLOv11m compared to 1.5 ms for YOLOv11n, indicating that the Medium variant requires substantially more computational resources and has a slower inference speed [22]. Therefore, although YOLOv11m demonstrated slightly better detection performance in this study, YOLOv11n provides a significant advantage in computational efficiency and inference speed, which should also be considered when selecting a model for practical deployment. The YOLOv11m model can be trained further and potentially outperformed YOLOv11n at a bigger gap as it is capable of extracting complex features and understanding cases that the YOLOv11n model might miss.

In conclusion, YOLOv11m is chosen not only due to its results in performance metrics such as mAP50, precision, and recall, but also for its growth potential that can place it further above YOLOv11n in future iterations.

C. *Importance of True Negatives*

Although the proposed system is primarily designed for single-class object detection (pothole detection), the concept of true negatives remains important in evaluating model reliability. A true negative occurs when the model correctly identifies a non-pothole region as not containing a pothole. While the model is trained to detect only one class, its performance is still influenced by its ability to avoid incorrectly classifying visually similar background objects as potholes.

In practical deployment, this is particularly important for reducing false alarms in the reporting system. For example, as shown in Fig. 10, a circular soil bed for trees was incorrectly detected as a pothole with high confidence. This demonstrates that although the model is trained specifically for pothole detection, it does not inherently understand the concept of "non-pothole" objects beyond the absence of learned pothole features. As a result, visually similar patterns in the environment may still trigger false positive detections.

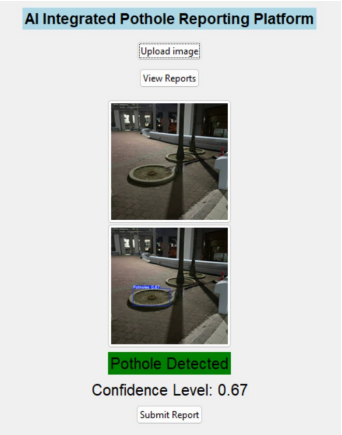

Fig. 10. False positive detected by the reporting system AI model

When true negatives (non-pothole instances) are not adequately represented during model training, the dataset can be considered imbalanced, which may introduce a bias in the model towards the positive class (potholes). This can cause the model to over-predict pothole instances, increasing the number of false positives when encountering non-potholes with visually similar structures. Consequently, this reduces the overall reliability of the system in real-world scenarios, where a wide variety of road and environmental textures may be present. Improving true negative performance contributes to reducing false positives, which is critical for maintaining the accuracy and credibility of the AI-based pothole reporting system.

IV. Conclusion and Future Works

In conclusion, this paper presented a comparative performance analysis of the YOLOv11m and YOLOv11n object detection models for an AI-based pothole detection and reporting system. Both models were trained and evaluated under identical configuration settings using the MWPD dataset and assessed using multiple performance metrics, including mAP50, precision, recall, precision–recall curves, and confusion matrix analysis. Experimental results demonstrated that YOLOv11m consistently outperformed YOLOv11n, achieving mAP50 score of 0.76692, precision of 0.76728, and recall of 0.7020, making it the preferred model for integration into the proposed reporting platform. While YOLOv11m offered slight accuracy improvements over YOLOv11n, YOLOv11n performed remarkably closely despite having substantially fewer parameters and computational requirements, demonstrating that the smaller model can achieve comparable detection performance with considerably lower computational requirements.

The selected YOLOv11m model was successfully integrated into a prototype reporting platform capable of automatically analysing user-submitted road images to detect the presence of potholes, displaying detection confidence level, and recording reports information. The developed system demonstrates the practical feasibility of incorporating deep learning-based object detection into municipal road reporting workflows, reducing the need for manual verification efforts while improving the efficiency, consistency, and reliability of pothole report validation. Furthermore, the proposed system supports the advancement of intelligent infrastructure management and contributes to Sustainable Development Goal (SDG) 9: Industry, Innovation and Infrastructure and SDG 11: Sustainable Cities and Communities through the application of AI technologies for safer and more sustainable transportation infrastructure.

Throughout the evaluation, it was observed that visually similar road features may occasionally be misclassified as potholes. This highlights the importance of including sufficient negative samples and diverse training data during model development. Although the selected model achieved satisfactory performance, the results indicate that further improvements can be achieved by using larger and more diverse datasets, extended training epochs, and additional computational resources. Future work may also investigate other YOLO variants or alternative object detection architectures, such as the newly released YOLOv26 which claims to be outperforming YOLOv11 in terms of mAP, refined Oriented Bounding Boxes (OBB) and instance segmentation [29]. In addition, future development may focus on real-time

deployment through web or mobile platforms and integration with Geographic Information Systems (GIS) to support practical implementation in smart city road maintenance applications.

ACKNOWLEDGMENT

Sincere appreciation is dedicated to Faculty of Electrical Engineering, Universiti Teknologi MARA (UiTM) for providing the academic environment and resources that has supported the development of this project. Special gratitude is extended to N. I. Author for continuous guidance, constructive feedback and technical advice given throughout the journey in completing this project.

REFERENCES

[1] Shtayat, S. Moridpour, B. Best, A. Shroff, and D. Raol, "A review of monitoring systems of pavement condition in paved and unpaved roads," *Journal of Traffic and Transportation Engineering (English Edition)*, vol. 7, no. 5, pp. 629–638, Aug. 2020, doi: 10.1016/j.jtte.2020.03.004

[2] "Ensure road safety with reinforced paved roads," *STRATA GLOBAL*, Jun. 12, 2025. https://www.strataglobal.com/glossary/paved-roads

[3] M. Golos, "What causes potholes?," *Tensar*, Oct. 20, 2022. https://www.tensarinternational.com/resources/articles/what-causes-potholes

[4] T. H. A. Rodzi, "Aplikasi MYJalan dibuka untuk aduan kerosakan jalan," *Berita Harian*, Aug. 22, 2023. [Online]. Available: https://www.bharian.com.my/berita/nasional/2023/08/1143511/aplikasi-myjalan-dibuka-untuk-aduan-kerosakan-jalan#google_vignette

[5] "Pelancaran Kempen MyJalan Kementerian Kerja Raya | Kementerian Kerja Raya." https://www.kkr.gov.my/ms/pelancaran-kempen-myjalan-kementerian-kerja-raya

[6] S. D. Reporter, "Potholes plaguing Malaysian roads: New solution found? | Sinar Daily," *Sinar Daily*, Sep. 13, 2024. [Online]. Available: https://www.sinardaily.my/article/221417/focus/national/potholes-plaguing-malaysian-roads-new-solution-found

[7] G. Gimino, "Potholes pose danger," *The Star*, Jun. 14, 2025. [Online]. Available: https://www.thestar.com.my/news/nation/2025/06/14/potholes-pose-danger

[8] S. Malik, K. Muhammad, and Y. Waheed, "Artificial intelligence and industrial applications-A revolution in modern industries," *Ain Shams Engineering Journal*, vol. 15, no. 9, p. 102886, Jun. 2024, doi: 10.1016/j.asej.2024.102886.

[9] I. B. Hassani, S. Benhida, N. Lamii, K. Oqaidi, A. Ouiddad, and S. Ghiadi, "From YOLO V1 to YOLO V11: comparative analysis of YOLO algorithm (review)," *International Journal of Power Electronics and Drive Systems/International Journal of Electrical and Computer Engineering*, vol. 16, no. 1, p. 450, Jan. 2026, doi: 10.11591/ijece.v16i1.pp450-462.

[10] E. H. Alkhammash, "A comparative analysis of YOLOV9, YOLOV10, YOLOV11 for smoke and fire detection," *Fire*, vol. 8, no. 1, p. 26, Jan. 2025, doi: 10.3390/fire8010026.

[11] A. Sharma, V. Kumar, and L. Longchamps, "Comparative performance of YOLOv8, YOLOv9, YOLOv10, YOLOv11 and Faster R-CNN models for detection of multiple weed species," *Smart Agricultural Technology*, vol. 9, p. 100648, Nov. 2024, doi: 10.1016/j.atech.2024.100648.

[12] F. Aldi, I. Nozomi, M. Hafizh, and T. Novita, "Comparative Analysis of YOLOv11 with Previous YOLO in the Detection of Human Bone Fractures," *Journal of Computer Networks Architecture and High Performance Computing*, vol. 7, no. 3, pp. 777–790, Jul. 2025, doi: 10.47709/cnahpc.v7i3.6051.

[13] Q. T. Nguyen, H. A. Vu, N. D. T. Ngan, and M. P. H. Ngoc, "Performance Analysis of YOLOv11 and YOLOv8 for Mixed Traffic Object Detection under Adverse Weather Conditions in Developing Countries," *arXiv (Cornell University)*, Jun. 2026, doi: 10.48550/arxiv.2606.12066.

[14] A. A. Murat and M. S. Kiran, "A comprehensive review on YOLO versions for object detection," *Engineering Science and Technology an International Journal*, vol. 70, p. 102161, Aug. 2025, doi: 10.1016/j.jestch.2025.102161.

[15] M. A. Rasee *et al.*, "AI-driven vision-based pothole detection for improved road safety," *Pertanika Journal of Science & Technology*, vol. 33, no. 3, Apr. 2025, doi: 10.47836/pjst.33.3.20.

[16] I. Almasri, D. Manasreh, and M. D. Nazzal, "AI meets ADAS: Intelligent pothole Detection for safer AV navigation," *Vehicles*, vol. 7, no. 4, p. 109, Sep. 2025, doi: 10.3390/vehicles7040109.

[17] L. V. Fortin, A. R. V. D. Santos, P. J. B. Cagud, P. R. Castor, and O. E. Llantos, "Eyeway: An Artificial Intelligence of Things pothole detection system with map visualization," *Procedia Computer Science*, vol. 251, pp. 216–223, Jan. 2024, doi: 10.1016/j.procs.2024.11.103.

[18] V. Vardhan, B. Avinash, G. Dharmalingam, D. Akash, K. Sivamma, and M. S. Rani, "AI based road condition monitoring pothole detection using YOLOv11 and deep learning with Python," *International Journal of Engineering Research and Science & Technology*, vol. 22, no. 2(1), pp. 752–760, Apr. 2026, doi: 10.62643/ijerst.v22.i2(1).2657.

[19] A. A. Murat and M. S. Kiran, "A comprehensive review on YOLO versions for object detection," *Engineering Science and Technology an International Journal*, vol. 70, p. 102161, Aug. 2025, doi: 10.1016/j.jestch.2025.102161.

[20] K. Maharana, S. Mondal, and B. Nemade, "A review: Data pre-processing and data augmentation techniques," *Global Transitions Proceedings*, vol. 3, no. 1, pp. 91–99, Apr. 2022, doi: 10.1016/j.gltp.2022.04.020.

[21] K. R. Kalla, B. B. Rath, N. K. Rayaguru, R. K. Bodda, A. Ganga, and P. K. Gouda, "Robust prediction of solar global tilted irradiance using gradient boosting and Kneedle-Based data splitting," in *2025 11th International Conference on Electrical Energy Systems (ICEES)*, IEEE, Aug. 2025, pp. 103–108. doi: 10.1109/icees67011.2025.11213272.

[22] R. Khanam and M. Hussain, "YOLOV11: An overview of the key architectural enhancements," *arXiv.org*, Oct. 23, 2024. https://arxiv.org/abs/2410.17725

[23] E. H. Alkhammash, "A comparative analysis of YOLOV9, YOLOV10, YOLOV11 for smoke and fire detection," *Fire*, vol. 8, no. 1, p. 26, Jan. 2025, doi: 10.3390/fire8010026.

[24] P. T. Nguyen, D. C. Huynh, L. D. Ho, H. A. Tran, and M. W. Dunnigan, 'Improving the YOLOV11 model for detecting plant diseases', *Journal of Engineering Science and Technology*, vol. 20, no. 5, pp. 1500–1515, 2025.

[25] G. M, B. S. D, D. S, and S. L, "A novel YOLOV11M-Based deep learning system for early detection and control of groundnut leaf diseases," *International Journal of Drug Delivery Technology*, vol. 16, no. 37s, May 2026, doi: 10.25258/ijddt.16.37s.11.

[26] Ultralytics, "Ultralytics YOLO11," *Ultralytics Docs*, Jun. 25, 2026. https://docs.ultralytics.com/models/yolo11#usage-examples

[27] Ultralytics, "YOLO Performance Metrics," *Ultralytics Docs*, Jul. 06, 2026. https://docs.ultralytics.com/guides/yolo-performance-metrics#visual-outputs

[28] Ultralytics, "Ultralytics YOLO11," *Ultralytics Docs*, Jun. 25, 2026. https://docs.ultralytics.com/models/yolo11#performance-metrics

[29] G. Jocher, J. Qiu, M. Liu, S. Lyu, F. C. Akyon, and M. E. Kalfaoglu, "Ultralytics YOLO26: Unified Real-Time End-to-End Vision Models," *arXiv.org*, Jun. 02, 2026. https://arxiv.org/abs/2606.03748