

# "Mind the Gap" - A Deep Learning Analysis of Pothole Detection

Itai Benyamin<sup>1</sup> and Idan Baruch<sup>1</sup>

<sup>1</sup>*Faculty of Electrical and Computer Engineering, Technion - Israel Institute of Technology*

February 4, 2025

## Abstract

This project evaluates several state-of-the-art object detection models on the task of pothole detection in road infrastructure, training them to identify potholes, and assessing their precision. Unlike existing pothole detection systems, it specifically examines the impact of simulated motion blur from moving vehicles and camera shake on model precision, while also exploring strategies to enhance robustness to noise and generalization through data augmentation. Furthermore, severity-based labels are added to a widely used annotated potholes dataset. The best performing models are then re-trained and evaluated on this more challenging task, demonstrating their ability to not only detect potholes but also classify their severity based on the potential risk to pedestrians and vehicles.

## 1 Introduction

Potholes are one of the most common and dangerous forms of road damage, posing significant risks to the safety of vehicles and pedestrians, while also contributing to high road maintenance costs worldwide. Traditional methods of detecting potholes typically involve manual inspections by road maintenance workers or the use of specialized vehicles equipped with sensors and cameras. These approaches, however, are often time-consuming, labor-intensive, and prone to human error. The ever growing computer vision and deep learning fields introduce object detection algorithms, which when done correctly, could be trained to detect such road hazards in a timely, more efficient manner.

Several studies have explored deep learning-based approaches for detecting road defects, including potholes. However, existing methods either lack severity classification, fail to account for real-world motion blur, or do not view the problem from a pedestrian or vehicle point-of-view.

Leading Kaggle notebooks [1, 2] demonstrate pothole detection using SSD and timm models, however lack comparison between different models. Tamagusko et al. [3] show that the customization of pre-trained deep learning models, in particular *You Only Look Once* (YOLO), can lead to impressive precision in detection potholes. However, this lacks methods to classify the potholes based on their severity.

This issue is addressed in recent work by Bhatia et al. [4], which introduces an algorithm that classifies potholes into five severity levels. But, it is designed for road quality analysis and does not account for potential degradation caused by real-world noise. The latter seems to be overlooked by others, such as Fan et al. [5], which despite the high accuracy of their model, rely on costly sensors and the design is not suitable for pedestrians' or vehicles' point-of-view.

An integral part of this work's novelty lies in its approach to handling noise, specifically motion blur, in the context of pothole detection. While Sayed et al. [6] address object detection under motion blur by modifying the internal architecture of detection models, they do not explore data augmentation as a means of improving robustness. In contrast, this study examines the impact of motion blur on pothole detection by introducing self-synthesized motion blur kernels and simulated camera shake. Additionally, it leverages Kornia [7] to implement data augmentation techniques that enhance model performance on images degraded by visual conditions.

To evaluate the effects of motion blur, multiple state-of-the-art object detection models are trained for the task of pothole detection. The models are then evaluated based on their performance under the test set with and without custom motion blur noise. This noise was generated using randomized kernel types and sizes, as well as deploying a set of widely used recorded camera shake kernels [8], to replicate common real-world conditions. In addition to detecting potholes, this work introduces a severity classification in order to train models to assess the threat of a given pothole to pedestrians and vehicles.

## 2 Methods

### 2.1 Object Detection

Object detection tasks are defined as locating and identifying items in an image or video. An object detection model usually marks each object it detects in an image with bounding boxes, a prediction for the object's class label, and a confidence score that the detected object belongs to the predicted class. This is in contrast to image classification, where a single label is given to an entire image. Generally, an object detection model processes an input image by extracting features through a Convolutional Neural Network (CNN) and then outputs bounding boxes along with their corresponding class labels and confidence scores.

#### 2.1.1 Evaluation Metrics

- **IoU** (Intersection over Union) measures the overlap between the predicted and ground-truth bounding boxes. It is defined as:

$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$

- **mAP@50** (mean Average Precision at 0.5 IoU) evaluates object detection performance by computing the average precision, where predictions with an IoU score above the threshold of 0.5 are considered true positives (TP), and those below the threshold are counted as false positives (FP). It is defined as:

$$\text{mAP}@50 = \frac{1}{N} \sum_{i=1}^N \text{AP}_i$$

where  $N$  is the number of classes, and  $\text{AP}_i$  is the Average Precision for class  $i$ .

- **FPS** (Frames Per Second) refers to the number of images a model can process in one second. The FPS of a given model is defined as:

$$\text{FPS} = \frac{1}{\text{Inference Time (seconds)}}$$

Since this project is primarily meant to be deployed on real-time video streams, FPS is chosen as the key metric for evaluating the model's inference speed.

### 2.2 Dataset

The dataset used for training and evaluating the models in this work is the *Annotated Potholes Dataset* from Kaggle [9]. It contains 665 road images from various locations with 1740 tagged potholes. Figure 1 shows a number of random example images taken from the dataset, alongside their respective annotations.



Figure 1: Five randomly chosen images from the dataset with the annotations of the potholes.

## 2.3 Revised Dataset

To address the absence of severity classification and annotation inaccuracies in the widely used dataset mentioned in section 2.2, an enhanced version of the original dataset is introduced [10]. The revised dataset includes the following key improvements:

- **Severity Levels** - each pothole is manually categorized into one of three severity levels based on the danger it poses to both pedestrians and vehicles:
  - **Minor Pothole** poses little to no threat to both pedestrians and vehicles.
  - **Medium Pothole** poses potential hazard to pedestrians but minor or non-threatening to vehicles.
  - **Major Pothole** poses a significant risk to both pedestrians and vehicles.
- **Dataset Enhancement** - errors in the original dataset have been corrected, and an additional 52 images from roads in Israel have been included, increasing dataset diversity and robustness.

This revised dataset improves both the accuracy of annotations and the ability to assess pothole severity, allowing more effective road maintenance and threat prioritization based on the potential damage that can be caused to pedestrians and cars.



Figure 2: Three images from the new dataset, and the distribution of the different classes in the dataset.

## 2.4 Motion Blur

In real-life applications, various types of noise can significantly affect the accuracy of the trained model, when it was only trained on images without any noise. In applications of detecting road hazards from a pedestrian or a vehicle point-of-view, a common occurring noise is motion blur. This noise, induced for example by moving cars or camera shake, is often modeled as a kernel convolved with the original image in addition to some added general noise:

$$I_{mb} = k * I + n \quad (1)$$

Here,  $I_{mb}$  is the motion blurred image,  $I$  is the original image,  $k$  is the motion blur kernel and  $n$  is the noise (usually modeled as white Gaussian noise). There are two different kinds of motion blur kernels which are used to add noise to the test set: Levin et al. [8] presents eight natural kernels in various shapes, achieved by recording the same scene with and without camera shake, then reconstructing the kernel to approximate the camera shake's point spread function (PSF). Examples for natural kernel patterns are presented in the first row of Figure 3.

The second class of kernels is the synthesized kernel which is artificially generated and can have various shapes such as uniform or ellipsoid. Unlike the natural kernels, in the synthesized kernels the blur strength is manually adjustable. Examples of uniform and ellipsoidal synthesized kernels are shown in the second and third row of Figure 3, presenting different levels of noise amplitudes  $a \in [0, 1]$  which controls the strength of blur.

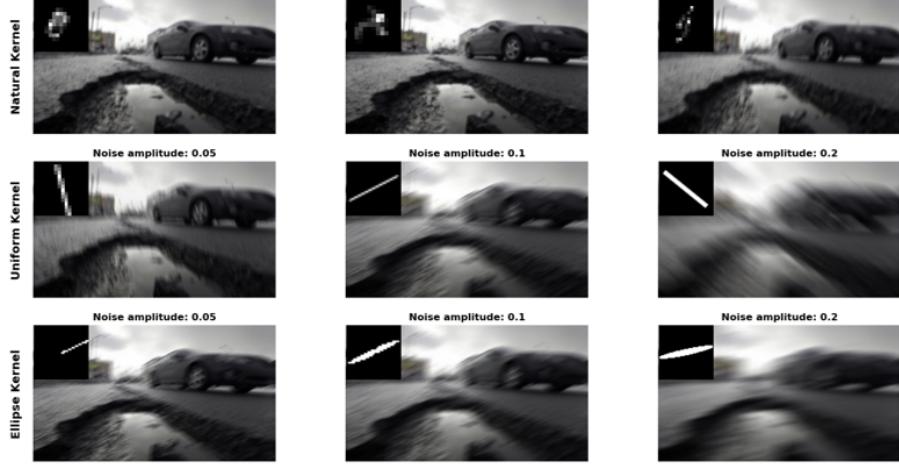


Figure 3: Example images of natural and synthesized (both uniform and ellipsoid) kernels in various shapes applied on an image. For the synthesized kernels, the blur strength is adjusted via the noise amplitude  $a$ .

In this work, custom kernels are employed in order to better approximate a real motion blur PSF and to ensure testing data is not influenced by the same augmentation used in training.

## 2.5 Data Augmentation

To cope with the motion blur noise, this study uses **Kornia** to apply the data augmentations presented in Table 1 on the train set. Then, the models trained on the augmented data were re-evaluated on the test set, with and without added motion blur noise.

Table 1: Data augmentations used to improve models' generalization and robustness to motion blur noise.

Augmentation	Parameters	Description
RandomMotionBlur	<code>kernel_size=(3,51), angle=(-180.0,180.0), direction=(-1.0,1.0), p=0.4</code>	Apply random motion blur.
RandomGaussianBlur	<code>kernel_size=(3,3), sigma=(0.1,2.0), p=0.3</code>	Apply random Gaussian blur to simulate out-of-focus blur.
RandomSharpness	<code>sharpness=(0.5,2.0), p=0.3</code>	Adjust the sharpness of the image to simulate varying levels of focus.
ColorJiggle	<code>brightness=0.2, contrast=0.2, saturation=0.2, p=0.2</code>	Apply a random transformation to the brightness, contrast, saturation, and hue.

These augmentations were chosen to help the model generalize better to different types of blur that might be encountered in real-world scenarios.

## 2.6 Models

This project compares five state-of-the-art object detection models. From **torchvision**, the models used are Faster R-CNN, RetinaNet, FCOS, and SSD. YOLOv8m was included additionally as it is the most used model in many different works. Faster R-CNN and RetinaNet utilize anchor-based approaches with region proposals, while FCOS is an anchor-free detector relying on dense predictions. SSD offers a balance between speed and accuracy by detecting objects at multiple scales directly from feature maps. YOLOv8m stands out for its high inference speed and real-time performance capabilities. In this project, hyperparameter tuning was conducted using **Optuna** to optimize training on torchvision models. A comprehensive search was performed across multiple hyperparameter spaces, including model preweight strategies, optimizer types, learning rates, momentum, and weight decay parameters. The study involved carefully selecting combinations of hyperparameters to achieve the best validation mAP@50.

### 3 Results

#### 3.1 Train Results

For each model, the training loss functions and the validation mAP@50 over 100 epochs is presented in Figure 4. The FCOS ResNet’s loss seem to converge into a local minimum, while the SSDLite MobileNet’s loss heavily oscillate, in correlation to both models achieving low mAP results. It can be seen, that the YOLOv8m model reached the best mAP@50 on the validation set, which correlates to its pre-weights performance on the COCO dataset.

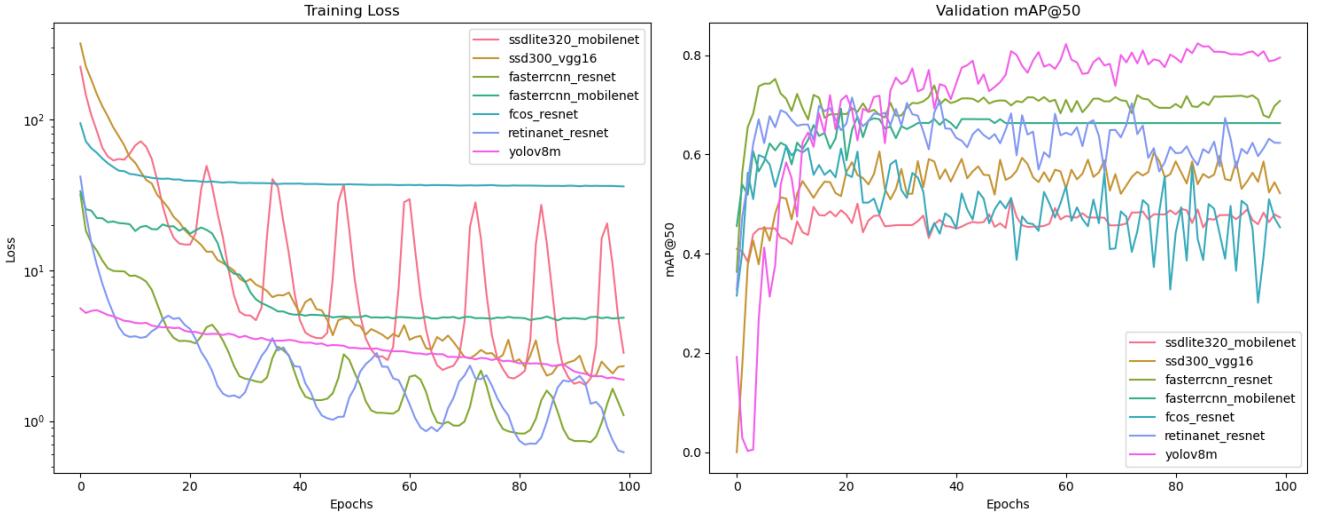


Figure 4: Training loss and validation mAP@50 over 100 epochs.

#### 3.2 Test Results

In Figure 5 (a), the results of running the listed models on a dataset without noise, while the models were trained on both a train set without augmentation, and the augmented training set. It shows that all models with ResNet as a backbone, increase their performance when being trained on the augmented train set compared to the regular train set. This is probably due to ResNet’s effective feature extraction due to its skip connections, which helps in preserving feature integrity even in noisy environments. All other models decrease their performance. This changes when using a test set which includes natural recorded camera shake kernel motion blur noise, as shown in Figure 5 (b). In this case, all models trained on the augmented train set reach better accuracy results, proving the effectiveness of data augmentation in coping with real scenario noise.

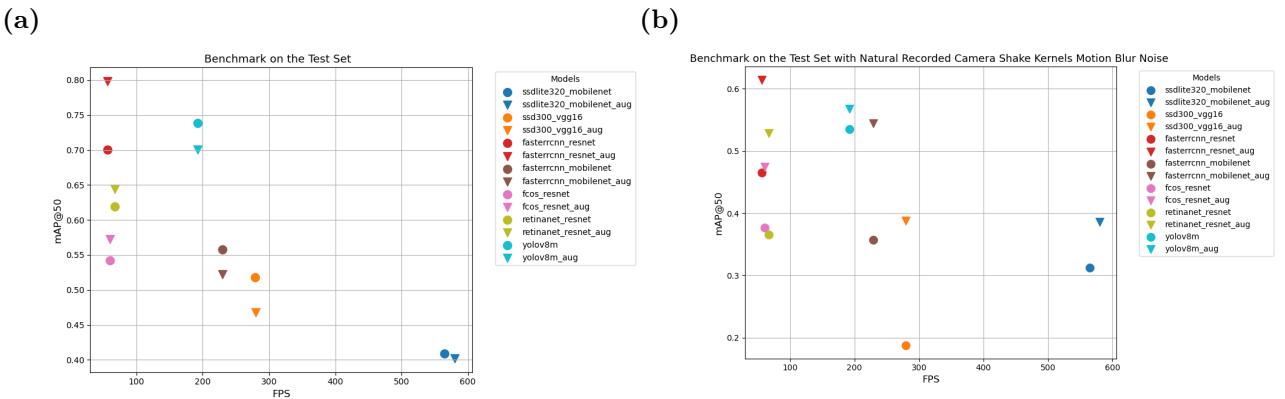


Figure 5: Models’ performance when trained on both augmented and regular train set, evaluated on a test set with (a) no additional noise, (b) added camera shake noise.

Testing the models on test sets with synthesized ellipse-kernel motion blur noise and gradually increasing the added noise from  $a=0.01$  to  $a=0.1$ , which is presented in Figure 6, shows that all models' accuracy decreases, as it's becoming harder to detect the potholes correctly. The model which turned out to be the most robust to the added noise is the YOLOv8m, even though it was only the second best model after Faster RCNN ResNet at lower noise addition. Also to be noted, is that in this case, the models which are trained with the augmented dataset perform significantly better than the non-augmented ones.

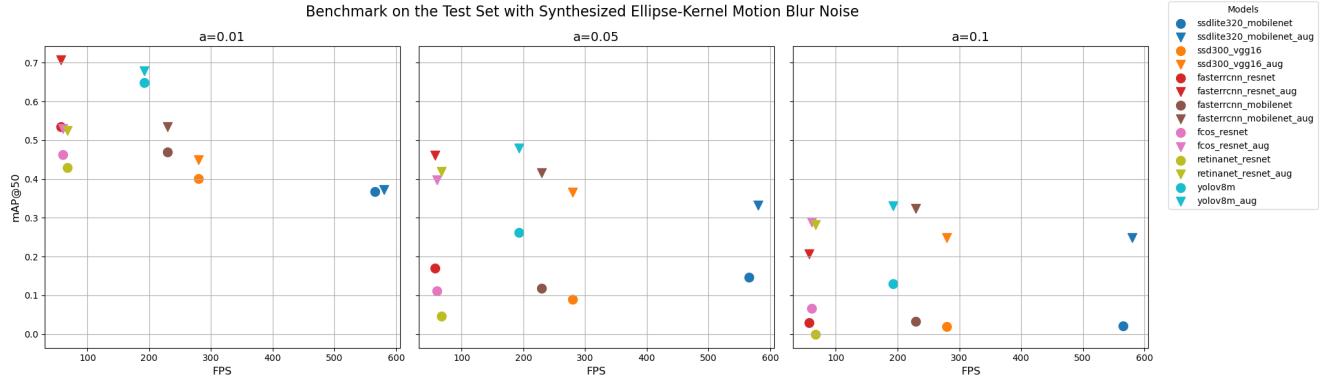


Figure 6: Models' performance when trained on both augmented and regular train set, evaluated on a test set with added ellipse-shaped synthesized motion blur of amplitude  $a \in \{0.01, 0.05, 0.1\}$  respectively.

Detailed information about the different combinations of models trained on regular/augmented train set and evaluated on test set with different kinds and levels of added noise is summarized in Table 2. In the last four rows are the best models trained on the revised dataset with the severity levels for each pothole. As expected from a more challenging task, both models shown significant drop in precision. The YOLOv8m model handles the new severity labels better than the Faster RCNN ResNet model, which had a bigger drop in its precision on the new dataset. This is likely due to YOLO's efficient one-stage design, which handles complex class distinctions better by directly predicting object locations and classifications. Dividing potholes by severity increased dataset complexity, where YOLO's anchor-free architecture and robust feature extraction proved more adaptable. In contrast, Faster RCNN ResNet, a two-stage detector, relies on precise region proposals and anchor configurations, making it less suited for fine-grained class variations, leading to its poor performance.

Table 2: Performance comparison of object detection models.

Model Name	FPS	Train Time (s)	Params (#)	Model Size (MB)	mAP@50 on Test Set							
					Without Noise	Uniform Motion Blur a=0.01	Uniform Motion Blur a=0.05	Uniform Motion Blur a=0.1	Ellipse Motion Blur a=0.01	Ellipse Motion Blur a=0.05	Ellipse Motion Blur a=0.1	Natural Camera Shake
ssdlite320_mobilenet	565.13	471.12	2,402,360	9.27	40.90%	36.06%	12.86%	2.77%	36.8%	14.66%	2.17%	31.2%
ssdlite320_mobilenet_aug	580.24	481.70	2,402,360	9.27	40.23%	36.35%	34.73%	29.0%	37.21%	33.15%	24.85%	38.61%
ssd300_vgg16	279.29	523.71	23,707,188	90.58	51.83%	42.13%	9.8%	0.99%	40.06%	8.92%	1.99%	18.77%
ssd300_vgg16_aug	279.60	537.66	23,707,188	90.58	46.81%	46.36%	41.2%	32.12%	44.94%	36.61%	24.93%	38.83%
fastercnn_resnet	56.43	2240.47	41,076,761	157.95	70.06%	55.67%	21.89%	2.65%	53.55%	17.0%	2.97%	46.55%
fastercnn_resnet_aug	56.43	2240.47	41,076,761	157.95	79.8%	72.79%	53.98%	32.16%	70.65%	46.09%	20.73%	61.42%
fastercnn_mobilenet	229.41	675.25	18,871,333	72.40	55.74%	50.88%	17.58%	5.79%	47.03%	11.88%	3.24%	35.72%
fastercnn_mobilenet_aug	229.41	675.25	18,871,333	72.40	52.24%	54.51%	48.4%	36.61%	53.52%	41.58%	32.42%	54.4%
fcos_resnet	60.37	2043.92	31,842,055	122.72	54.24%	46.42%	14.86%	4.48%	46.25%	11.22%	6.57%	37.71%
fcos_resnet_aug	60.37	2043.92	31,842,055	122.72	57.26%	55.52%	49.35%	34.89%	52.91%	39.81%	28.94%	47.41%
retinanet_resnet	67.16	2100.74	36,127,286	138.88	61.88%	42.12%	2.26%	9.91%	42.9%	4.64%	0.0%	36.54%
retinanet_resnet_aug	67.16	2100.74	36,127,286	138.88	64.44%	52.22%	43.45%	29.95%	52.53%	41.95%	28.2%	52.86%
yolov8m	192.31	632.5	25,856,899	98.76	73.85%	68.79%	24.25%	11.17%	64.95%	26.19%	12.94%	53.5%
yolov8m_aug	192.31	633.73	25,856,899	98.76	70.07%	67.23%	50.67%	39.38%	67.82%	47.89%	33.04%	56.8%
yolov8m_severity	192.31	663.86	25,858,057	98.77	49.15%	45.44%	19.6%	7.91%	47.5%	19.2%	9.48%	35.77%
yolov8m_severity_aug	192.31	663.81	25,858,057	98.77	48.79%	46.0%	34.87%	25.63%	43.96%	31.83%	21.76%	36.94%
fastercnn_resnet_severity_aug	54.82	2359.93	41,087,011	157.99	34.69%	34.29%	31.67%	24.06%	36.77%	28.8%	18.5%	33.7%
fastercnn_resnet_severity	54.96	2326.27	41,087,011	157.99	35.56%	22.59%	9.56%	0.7%	24.79%	7.81%	1.75%	19.86%

### 3.3 Performance Analysis

**YOLOv8m** achieves the highest mAP@50 (around 0.75) while maintaining a respectable FPS. The highest FPS (around 500+) is achieved by **SSDLite320 MobileNet** but at a sacrifice of accuracy (around 0.45 mAP). This model is ideal for applications requiring ultra-fast inference at the expense of detection quality. **RetinaNet ResNet** and **Faster RCNN ResNet** strike a balance between accuracy and speed, with mAP@50 around 0.60 and decent FPS values (100-200). These could be good general-purpose models, while **FCOS ResNet** shows relatively poor performance both in terms of mAP and

FPS, making its usage less appealing. Moreover, **Faster RCNN MobileNet** and **SSD300 VGG16** have moderate FPS (200-300) but exhibit lower accuracy compared to other models.

### 3.4 Recommendations

For applications such as autonomous driving, where strong motion blur noise might show and fast inference times are needed, **YOLOv8m** seems to be the best choice of model. For applications where the noise bar is lower and a high accuracy model is required, such as guidance systems for the visually impaired, **Faster RCNN ResNet** is outperforming the other models. For real-time systems prioritizing speed, **SSDLite320 MobileNet** has the fastest inference time and it is the smallest model to deploy.

## 4 Ethics Statement

To ensure that any system built using the models and methods described in this project benefits society, it is important to consider its impact on different people and groups. The following sections discuss the key ethical considerations of this system, including identifying stakeholders, how they should be informed, and who is responsible for providing the explanations.

To leverage modern tools and technologies, as well as to avoid potential bias, ethical questions are first presented to a Large Language Model (LLM) and later manually reviewed. The LLM chosen for this task is ChatGPT-4 Turbo. Before gathering LLM responses, a description of this project is provided as a prompt.

### 4.1 LLM-Generated Responses

**Prompt:** List three types of stakeholders that will be affected by the project.

**Answer:**

- **Government and Road Maintenance Authorities:** Responsible for road safety and infrastructure quality.
- **Vehicle and Pedestrian Users:** End-users directly impacted by potholes and their detection.
- **Technology Developers and Researchers:** Work on building, deploying, and refining detection systems.

**Prompt:** What will an explanation given to each stakeholder look like?

**Answer:**

This project aims to improve road safety and infrastructure by detecting potholes in real time, even under challenging conditions like motion blur, using advanced object detection models. For government and maintenance authorities, the system provides a proactive solution for prioritizing repairs and resource allocation. For vehicle users and pedestrians, it ensures safer travel by minimizing accidents and damages caused by potholes. For developers and researchers, it pushes the boundaries of object detection technology in real-world scenarios, offering valuable insights and innovations.

**Prompt:** Who is responsible for giving the explanation to each stakeholder?

**Answer:**

The project manager or a designated representative should explain the technology to government and maintenance authorities, highlighting societal and economic benefits. The marketing or public relations team should communicate with vehicle users and pedestrians, focusing on safety and practical advantages. The technical lead or researchers should engage with developers and researchers, ensuring the scientific and technical aspects are effectively conveyed to foster further advancements.

### 4.2 Reflection on LLM Responses

The LLM effectively identified relevant stakeholders, as well as key aspects of explanations and responsibilities. However, its responses focus on this project's advantages while overlooking potential issues such as detection and training biases. Detection inaccuracies and model performance must be transparent to

stakeholders, as inaccurate results can mislead and potentially harm users and organizations relying on the system. Furthermore, training biases may reduce the model’s accuracy in areas or countries with different road infrastructures than those used during training. To ensure ethical deployment of the system, any limitation of it should be disclosed by the project manager or those who deploy these technologies.

## 5 Conclusion

In this work, different deep learning models were evaluated regarding their suitability for precise and fast pothole detection. The training loss and validation mAP@50 showed weaknesses of some of the models, while YOLOv8m and Faster RCNN ResNet showed the best performance. Specifically, the impact of noise induced by real world scenarios, such as moving car and camera shake, was investigated by using custom noise kernels of various shapes and blur levels. The results show a significant negative impact of added noise on the performance of all investigated models. The method of using data augmented during training was employed to enhance the robustness of the models when evaluated on the test set with different added noises. Corresponding to theoretical expectations, this method significantly increased the precision of all investigated models with YOLOv8m showing the best robustness to high levels of noise. In order to contribute to the community, the used Kaggle dataset was revised by introducing three different pothole classes base on their severity, adding new images and correcting previous inaccuracies. Future research can use deploy this work’s results in guidance systems for vehicles and pedestrians, without the need for expensive measurement setup.

## References

- [1] Sudhanshu2198, “Real-time pothole detection using ssd,” Kaggle, 2024, accessed: 4 Feb. 2025. [Online]. Available: <https://www.kaggle.com/code/sudhanshu2198/real-time-pothole-detection-using-ssd/notebook>
- [2] Bambotims, “Pothole detection,” Kaggle, 2024, accessed: 4 Feb. 2025. [Online]. Available: <https://www.kaggle.com/code/bambotims/pothole-detection/>
- [3] T. Tamagusko and A. Ferreira, “Optimizing pothole detection in pavements: A comparative analysis of deep learning models,” *Engineering Proceedings*, vol. 36, no. 1, p. 11, 2023.
- [4] S. Bhatia and R. Greer, “Classifying pothole severity with convolutional neural networks,” in *2023 IEEE MIT Undergraduate Research Technology Conference (URTC)*. IEEE, 2023, pp. 1–4.
- [5] R. Fan, U. Ozgunalp, B. Hosking, M. Liu, and I. Pitas, “Pothole detection based on disparity transformation and road surface modeling,” *IEEE Transactions on Image Processing*, vol. 29, pp. 897–908, 2019.
- [6] M. Sayed and G. Brostow, “Improved handling of motion blur in online object detection,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 1706–1716.
- [7] D. P. E. R. E. Riba, D. Mishkin and G. Bradski, “Kornia: an open source differentiable computer vision library for pytorch,” in *Winter Conference on Applications of Computer Vision*, 2020. [Online]. Available: <https://arxiv.org/pdf/1910.02190.pdf>
- [8] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman, “Understanding and evaluating blind deconvolution algorithms,” in *2009 IEEE conference on computer vision and pattern recognition*. IEEE, 2009, pp. 1964–1971.
- [9] A. Rahman and S. Patel, “Annotated potholes image dataset,” 2020. [Online]. Available: <https://www.kaggle.com/dsv/973710>
- [10] I. Baru, “Annotated potholes with severity levels,” 2025, accessed: 2025-02-04. [Online]. Available: <https://www.kaggle.com/datasets/idanbaru/annotated-potholes-with-severity-levels>