applied sciences

Review

Comparison of CNN-Based Models for Pothole Detection in Real-World Adverse Conditions: Overview and Evaluation

Maroš Jakubec , Eva Lieskovská *, Boris Bučko and Katarína Zábovská

University Science Park UNIZA, University of Žilina, Univerzitná 8215/1, 010 26 Žilina, Slovakia; maros.jakubec@uniza.sk (M.J.); boris.bucko@uniza.sk (B.B.); katarina.zabovska@uniza.sk (K.Z.)
* Correspondence: eva.lieskovska@uniza.sk

Abstract: Potholes pose a significant problem for road safety and infrastructure. They can cause damage to vehicles and present a risk to pedestrians and cyclists. The ability to detect potholes in real time and with a high level of accuracy, especially under different lighting conditions, is crucial for the safety of road transport participants and the timely repair of these hazards. With the increasing availability of cameras on vehicles and smartphones, there is a growing interest in using computer vision techniques for this task. Convolutional neural networks (CNNs) have shown great potential for object detection tasks, including pothole detection. This study provides an overview of computer vision algorithms used for pothole detection. Experimental results are then used to evaluate the performance of the latest CNN-based models for pothole detection in different real-world road conditions, including rain, sunset, evening, and night, as well as clean conditions. The models evaluated in this study include both conventional and the newest architectures from the region-based CNN (R-CNN) and You Only Look Once (YOLO) families. The YOLO models demonstrated a faster detection response and higher accuracy in detecting potholes under clear, rain, sunset, and evening conditions. R-CNN models, on the other hand, performed better in the worse-visibility conditions at night. This study provides valuable insights into the performance of different CNN models for pothole detection in real road conditions and may assist in the selection of the most appropriate model for a specific application.

Keywords: pothole detection; adverse conditions; R-CNN; YOLO; object detection

check for updates

Citation: Jakubec, M.; Lieskovská, E.; Bučko, B.; Zábovská, K. Comparison of CNN-Based Models for Pothole Detection in Real-World Adverse Conditions: Overview and Evaluation. *Appl. Sci.* **2023**, *13*, 5810. https://doi.org/10.3390/app13095810

Academic Editors: Omneya Attallah and Kadir Sabancı

Received: 24 March 2023
Revised: 2 May 2023
Accepted: 4 May 2023
Published: 8 May 2023

1. Introduction

1.1. Problem Description and Motivation

Road transport plays a vital role in the economic development of a country, connecting communities and businesses while providing access to educational and employment opportunities, as well as healthcare and social services [1]. However, with the fast economic expansion and technological advancements in recent years, the quality of the transportation system has been affected. Ultimately, as road conditions deteriorate, traffic resilience weakens [2]. One of the most significant issues faced by drivers is the presence of potholes on roads. These not only cause damage to vehicles but also pose a risk to driver safety. According to a study by the AAA in 2016, potholes cost U.S. drivers USD 15 billion over the past five years in vehicle repairs, equivalent to nearly USD 3 billion per year [3].

Potholes can be caused by a variety of environmental factors, such as severe weather, high traffic loads [4], and heavy vehicles, as well as poor construction methods and lack of proper maintenance [5]. These issues are particularly difficult to address as they are often unexpected and require constant monitoring to ensure the road infrastructure [6] is in good condition. To mitigate the risk posed by potholes, a system that can detect and communicate their presence to maintenance services or other drivers could be implemented. This would not only improve driver safety but also assist relevant authorities in determining which potholes pose the greatest risk and need to be repaired as soon as possible [7]. The development of a variety of technologies that enable the collection, processing, and transfer of data in real time between fixed traffic devices, moving vehicles, and data centres has opened up a wide range of new possibilities for Intelligent Transport Systems (ITSs) [8].

The development of intelligent pothole detection technology is crucial to address issues with road traffic. However, in natural settings, the detection of potholes can be hindered by adverse weather conditions, such as rain, snow, and fog, as presented in [9–11]. On rainy days, potholes may be hidden under puddles or resemble puddles. Water on the windshield of a car can obstruct the driver's field of view and make it difficult to detect any damage to the road. The reduced visibility induced by the fog increases the likelihood that the car will sustain damage from potholes. Additionally, even in favourable conditions, there is still the possibility of erroneous detection. The results of the pothole detection in good-visibility conditions are displayed in Figure 1. On a sunny day, it is possible to observe instances of false detection, such as the reflection of a pothole on a car hood or a little road patch, denoted in green.

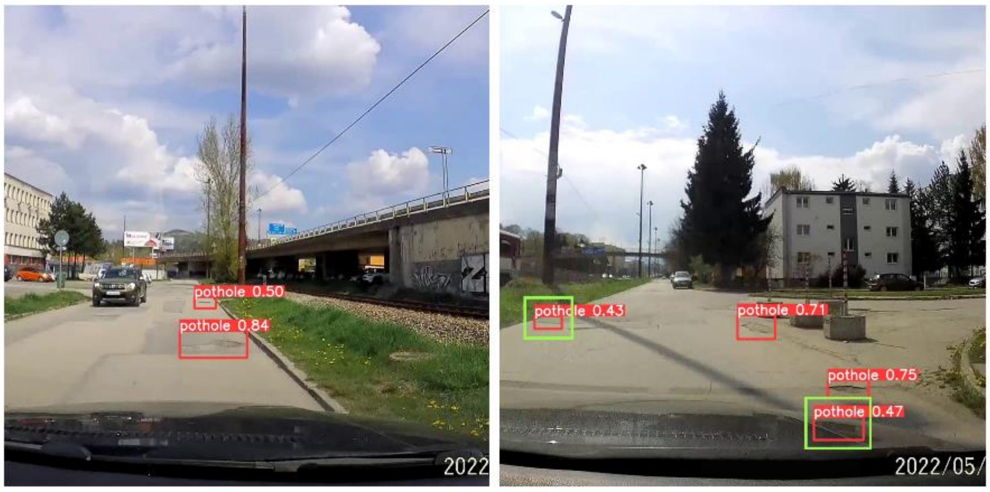

Figure 1. Detection outcomes under clear weather conditions.

Pothole detection in real conditions is a complex task that involves several factors, including the type of road surface, the condition of the road, and the traffic volume [12]. One of the main challenges in pothole detection is the ability to accurately identify and locate potholes in real time. Regardless of the approach used, pothole detection in real conditions requires a combination of hardware and software to accurately identify and locate potholes. This includes the use of advanced sensors, sophisticated algorithms, and machine learning techniques to process and analyse the data collected. Additionally, it is important to consider the environmental conditions, such as weather and lighting, which can affect the accuracy of pothole detection [13]. It is expected that with the continued development of sensor technology and machine learning techniques, pothole detection will become increasingly accurate and reliable in the future.

1.2. Pothole Detection

The location of road potholes may be determined in a number of different ways. Low-cost methods based on accelerometers measuring vibration can have a high degree of accuracy. Although their performance is independent on visibility conditions, their reaction time may be slow, and the vehicle must drive over a pothole to make detection [14–16].

Three-dimensional reconstruction techniques using laser [17,18] or stereo vision [19–21] can be utilized to model point cloud representations of road inconsistencies. 3D lasers based on the principle of transmitting light pulses to the target object/surface can detect objects in low visibility but come at a higher cost. Stereo vision systems can be susceptible to vibrations and need several camera alignments as well as significant surface reconstruction calculations.

Deep-learning-based object detection has been utilized in research to perform accurate and quick pothole detection. Object detection is a technique used to identify and classify objects within an image while determining their location. It is a critical challenge in the field of computer vision, and it has been the subject of intensive research and development over the course of the last several years. With the rapid development of deep learning, significant advancements have been made in improving object recognition performance using deep neural models. Among these models, Convolutional Neural Networks (CNNs) are considered one of the most innovative breakthroughs in image processing. One of the key advantages of using CNNs for object detection is that they can be trained end to end on large datasets, allowing us to learn complex features and representations of objects.

Region-based CNNs (R-CNNs) were among the first deep-learning-based object detection methods, proposing a CNN to extract features from input images and Support Vector Machines (SVMs) to classify objects [22]. However, their slow processing speed limits their practical use. Fast R-CNN [23] was later introduced to improve R-CNN by adding a Region-of-Interest (RoI) pooling layer that reduced processing time and allowed for end-to-end training of the network. You Only Look Once (YOLO) [24], a single-stage architecture, significantly improved detection accuracy, making it comparable to and, in some cases, even superior to R-CNN models. Providing significant accuracy and speed of inference, the YOLO models and their modifications appeared most often in research works regarding pothole detection. In addition, Faster R-CNN was also deployed for improved visual representation and detection accuracy. Although inference speed is a limitation of Faster R-CNN compared to newer versions of YOLO, there has been significant development of models in the R-CNN family recently.

In addition to the aforementioned architectures, EfficientDet [25], Single-Shot Detector (SSD) [26], and RetinaNet [10] are other models used for pothole detection. EfficientDet uses a compound scaling method to optimize the detection of objects at various sizes and scales. It achieves high accuracy with fewer parameters, making it a more efficient architecture for real-time applications. Similarly, RetinaNet uses a novel loss function to address the issue of class imbalance and improve the detection of smaller objects. SSD, on the other hand, simplifies the object detection pipeline by eliminating the region proposal step, resulting in faster detection times. A detailed overview of works related to pothole detection is given in Section 2.

Despite the progress that has been made in pothole detection using CNNs, there are still several challenges that need to be addressed to make this technology more practical and effective [27–29]. These include the need for larger and more diversified training datasets and algorithms capable of detecting potholes in real time and operating in a range of real-world settings, such as changing lighting and weather conditions.

The motivation of this paper is to provide a comprehensive overview of the different models available and to compare their performance in pothole detection in a complex environment through experiments. To the best of our knowledge, a comparison of pothole-detection-based systems concerning heterogeneous weather conditions is currently lacking. The main contributions of this paper are summarized as follows:

- An overview of contemporary neural networks employed in the pothole detection task and evaluation of the effectiveness of Fast-, Faster-, Mask-, Cascade-, Sparse R-CNN, and YOLO versions 3, -v4, -v5, -v6, and -v7 computer vision models for the pothole detection. The results of the study can provide valuable information for future research and can aid in the development of more accurate and reliable pothole detection systems.
- Comparison of models' performance in terms of detection accuracy under different weather conditions and exploring the capability of models dealing with adverse weather and light conditions. Although YOLO architectures perform detection with significant accuracy and speed, R-CNN models may handle the very low-visibility detection more successfully.

- Determining the limitations of existing deep neural networks for pothole detection and identifying techniques by which future research could improve pothole detection performance under adverse visual conditions.

Detecting potholes in adverse conditions and achieving a high level of accuracy at the same time is quite a difficult task, as indicated above. The main challenge in solving this problem is to work with the data collected under adverse conditions and to select an appropriate algorithm for detection. This work was carried out in six steps: firstly, a survey of existing methods for detecting road damage is reviewed and discussed in Section 2. Section 3 reviews the state-of-the-art algorithms for object detection. Section 4 describes the methodology used in this study, the data acquisition method, and the evaluation metrics. Section 5 discusses the experimental results and performs a comparative analysis between the state-of-the-art systems. In Section 6, we discuss the experiments, the benefits and limitations of the research, and recommendations for future research.

2. Related Work on Pothole Detection

Although the use of deep CNNs for pothole detection has been studied for several years [7,11,25,27,29,30], efficient algorithms for this task are still limited. In many scenarios, the pothole object is relatively small compared to the size of the road image, making it challenging to train a CNN on a high-resolution image due to the large amount of memory required. In a recent study by Pena-Caballero et al. [9], the performance of object identification and semantic segmentation algorithms was evaluated based on detection time and overall system accuracy. The results indicated that while segmentation algorithms had high accuracy, they often came with an increase in computational complexity. YOLO version 3 (v3) was found to be faster and achieve a better mean Average Precision (mAP) than YOLOv2.

Park et al. [30] presented a method for automated pothole detection based on YOLO models. Three YOLO models (YOLOv4, YOLOv4-tiny, and YOLOv5) were used in the training and testing phases, with a dataset of 665 pothole images divided into training, validation, and testing subsets. The results showed that the mAP@0.5 values of YOLOv4, YOLOv4-tiny, and YOLOv5 were 77.7%, 78.7%, and 74.8%, respectively, with YOLOv4-tiny having the best performance in detecting potholes. This study has limitations, such as low accuracy in detecting small, distant potholes and a lack of testing under poor weather and lighting conditions.

W. Ye et al. [27] presented a pothole detection method based on CNN pre-pooling. The dataset comprised 400 raw pothole images collected from different pavements under varying lighting conditions, cropped into 96,000 small images for training or testing. The proposed method utilized a pre-pooling layer in the CNN architecture, improving accuracy compared to conventional CNN. The authors analysed the proposed method's advantages in terms of accuracy, robustness to different lighting and pavement conditions, and superiority over conventional methods (e.g., Sobel edge detection and K-means cluster analysis).

Ahmed [11] performed an extensive evaluation of various object detection architectures, including YOLOv5 models (for three different model sizes) with ResNet101 backbone, YOLOR, Faster R-CNN with ResNet50, VGG16, MobileNetV2, and InceptionV3 backbones. The author also proposed a modified VGG16 (MVGG16) architecture, which effectively reduced computational costs while retaining detection accuracy. Upon comparison, Faster R-CNN with ResNet50 achieved the highest precision rate of 91.9% and an inference time of 0.098 s for larger images. YOLOv5, on the other hand, offered the fastest inference speed with an inference time of 0.009 s but at the cost of reduced accuracy, making it more suitable for real-time applications.

H. Chen et al. [29] proposed a pothole detection method based on localization and partial-classification CNNs. The proposed method comprised two subnetworks: Localization Network (LCNN) and the Part-based Classification Network (PCNN). The LCNN utilized a high recovery network to identify candidate regions, and the PCNN performed classification on the candidates. The authors used two different image sizes as inputs to the network to balance accuracy and efficiency. The results showed that the accuracy, precision, recall, and F1-score were 95.0%, 95.2%, 92.0%, and 93.6%, respectively, with the proposed method outperforming existing methods. The proposed method has the potential for scalability to detect not only potholes but also other road defects, but its performance may degrade with low-resolution input data. The authors suggest using scale selection and cascading techniques to further improve performance.

Heo et al. [7] proposed improvements in YOLOv4 Tiny through the integration of two multi-scale feature modules, such as the Spatial Pyramid Pooling (SPP) and Feature Pyramid Network (FPN) with CSPDarknet53-tiny backbone. Additional modules make up for the loss of spatial information by using a series of max-pool filters and extracting features of varying sizes (in SPP) and combining high-level feature maps from the deep convolutional layers with low-level features from the shallow layers (in FPN). The proposed SPFPN-YOLOv4 tiny model outperformed YOLOv2, YOLOv3, and YOLOv4 tiny, not only in mAP@.5 but also in frames per second rate (FPS).

Table 1 provides a summary of recent pothole detection systems. It is important to note that individual pothole detection systems may differ in the road damage dataset and hardware used, as well as in the training settings.

Table 1. A summary of recent pothole detection systems (DA—data augmentation).

References	Year	Enhancement Techniques	Model	Image Resolution	Inference Speed [FPS]	mAP@.5 [%]	mAP@ [0.5:0.95] [%]
Maeda et al. [31]	2018		SSD using Inception V2	300 × 300	16	–	–
			SSD MobileNet	300 × 300	33	–	–
Pena-Caballero et al. [9]	2020		SSD300 MobileNetV2	300 × 300	–	45.10	–
			YOLOv2	–	–	90.00	–
			YOLOv3	–	–	98.82	–
Chen et al. [29]	2020	The system contains localization and part-based classification networks.	LACNN	352 × 244 1760 × 1120	49	–	–
Ahmed [11]	2021	Incorporation of dilated convolution (MVGG16). DA: scaling, colour adjustments, rotation, mosaic	YOLOR-W6	1774 × 2365	31	–	44.60
			Faster R-CNN: MVGG16	1774 × 2365	21	–	45.40
			YOLOv5 (Ys)	1774 × 2365	111	–	58.90
			Faster R-CNN: ResNet50	1774 × 2365	10	–	64.12
Lin et al. [32]	2021	Two cameras with different fields of view. A tracking-via-detection	YOLOv3	416 × 416	35	71.00	–
Park et al. [30]	2021		YOLOv5s	720 × 720	–	74.80	–
			YOLOv4	720 × 720	–	77.70	–
			YOLOv4-Tiny	720 × 720	–	78.70	–

Table 1. *Cont.*

References	Year	Enhancement Techniques	Model	Image Resolution	Inference Speed [FPS]	mAP@.5 [%]	mAP@ [0.5:0.95] [%]
Salaudeen et al. [25]	2022	ESRGAN for image quality improvement	EfficientDet		–	39.00	26.00
			YOLOv5		–	46.00	32.00
Ramesh et al. [33]	2022	DA: flipping, 90° rotation, 20% crop	YOLOv5	–	–	93.70	–
Salcedo et al. [34]	2022	DA: flipping, warping, blurring, zooming	EfficientDet D1	–	–	63.00	–
			YOLOv5x	–	–	69.00	–
			YOLOv5l	–	–	74.00	–
Heo et al. [7]	2023	Multi-scale detection network proposed as a combination of SPP and FPN with CSPDarknet53-tiny (SPFPN) DA: gamma regulation, flipping and scaling	YOLOv4 tiny	–	–	72.70	–
			YOLOv2	–	–	74.80	–
			YOLOv3	–	–	77.80	–
			SPFPN-YOLOv4 tiny	–	–	79.60	–

3. Current State of Object Detection Algorithms

The purpose of detection networks is to locate objects in an image, often by defining an axially aligned bounding box that is centred on the object and to categorize that object at the same time. This is accomplished by specifying an axially aligned bounding box that is centred on the object. When there is sufficient overlap between an object's estimated location and the bounding box that a human has drawn around the object, the object's estimated location is accurate (ground truth). In this part of the article, several of the most well-known models for object recognition in images are going to be contrasted with one another in respect of their inner structure. The following two categories can be used to roughly classify these models:

- Two-stage—The operation of these models is divided into two phases. In the first phase, the so-called area design takes place. In these regions, predictions are then computed using the image classification model, and object classification is performed according to the results. This method is relatively slow because this procedure is applied to each proposed region. Representatives of these models include models from the R-CNN family.
- Single-stage—These models perform object localization and classification in a single pass through the neural network. This makes them significantly faster—they can perform real-time detections. Typical representatives include YOLO models and their variants.

3.1. Region-Based CNN

The original R-CNN model [22] was introduced in 2014, and it has been a crucial development in object detection systems, providing higher accuracy than other approaches. The R-CNN model proposes several RoIs in an image and performs convolution on these regions, which is a departure from the CNN approach of handling multiple regions. R-CNN uses a heuristic search technique to identify congested locations and the convolutional network receives these scaled regions of interest and generates a feature matrix. The presence of an object is then assessed via SVM using this feature matrix. In the final stage, the offset region is estimated to fine-tune and correct the position of the bounding box.

The main drawback of the R-CNN model is its time-consuming nature. Learning a neural network can be very time consuming, as each image may require the classification of up to 2000 distinct regions of interest, making real-time recognition virtually impossible. Additionally, the algorithm for selective search does not undergo any form of learning and is, hence, immutable, not learning to provide more appropriate options over time. This section will further provide an overview of the successors of R-CNN.

3.1.1. Fast R-CNN

The Fast R-CNN [23] was developed to address the limitations of the R-CNN. In this model, the complete input image is given to the CNN, which generates a feature matrix. This matrix is then used to identify the RoIs, which are scaled to the same aspect ratio and passed through an RoI pooling layer. A fully connected network is then used to classify the object and evaluate the bounding box. The Fast R-CNN model is faster than the R-CNN model, as only two convolution rounds are performed on the entire image. The learning process of the network is 10-times faster while consuming the same amount of hardware resources.

3.1.2. Faster R-CNN

The Faster R-CNN model [35] is another addition to the R-CNN family and is designed for real-time object classification. Unlike the Fast R-CNN and R-CNN models, the Faster R-CNN model replaces the selective search technique with a Region Proposal Network (RPN) to locate the specific location of an object. The input image is sent through the convolutional network, and the RPN is used to identify the RoIs. The RoI Pooling layer is then used to scale the RoIs, and the final step of object categorization and the bounding box refinement process is identical to that of the Fast R-CNN model. The Faster R-CNN model is up to ten-times faster than the Fast R-CNN model, making it suitable for real-time applications.

3.1.3. Mask R-CNN

Mask R-CNN is a deep learning model that extends the Faster R-CNN architecture by adding another output, the object mask. This model was introduced by He et al. [36] in 2017 and has since become a popular model for object detection and semantic segmentation tasks.

One of the key innovations of Mask R-CNN is the use of the RoI Align layer, which replaces the RoI Pool layer used in Faster R-CNN. The RoI Align layer helps to prevent misalignment at the pixel level, which leads to a more precise spatial distribution of the object. In Mask R-CNN, the two-stage procedure is maintained, with the same RPN being used in the first stage. In the second stage, Mask R-CNN outputs not only the class and bounding box predictions but also a binary mask for each ROI. This mask represents the spatial distribution of the object, and its pixel-to-pixel correspondence provides a natural way to extract the spatial structure of the mask. However, one of the main limitations of Mask R-CNN is its high computational cost, as it requires additional computations to generate the object masks.

3.1.4. Cascade R-CNN

Cascade R-CNN is an object detection model that improves the performance of the Faster R-CNN model. The model was introduced by Cai and Vasconcelos [37] in 2018 and addresses the issue of missed detections and false positives in Faster R-CNN by adding a series of detection heads with increasing Intersection Over Union (IoU) thresholds to the Faster R-CNN model. In Cascade R-CNN, each detection head is trained with a different IoU threshold, and the output of one detection head is used as the input for the next head. This cascading approach helps to reduce the number of false positives and improve the accuracy of object detection.

However, one of the main limitations of Cascade R-CNN is its increased complexity, as it requires multiple detection heads to be trained and deployed, which increases the computational cost and reduces the inference speed. In addition, the model may not perform well on smaller objects, as it relies on the output of previous heads, which may not detect small objects accurately.

3.1.5. Sparse R-CNN

Sparse R-CNN is an object detection model that is based on a predefined number of proposal boxes, which allows for more efficient processing of the input image. The model was introduced by Sun et al. [38] in 2021 and is designed to address the limitations of existing object detection models that rely on dense candidate boxes or region proposal networks. Instead of using dense candidate boxes, as used in single-stage architectures, or dense-to-sparse candidate boxes obtained using a RPN in two-stage architectures, the Sparse R-CNN utilizes learnable proposal boxes that are updated iteratively during training. This approach reduces the computational complexity of the model, allowing for faster processing times. Sparse R-CNN uses an FPN based on the ResNet backbone, along with a dynamic instance interactive head. The authors showed that the Sparse R-CNN model performs comparably to well-established detector baselines, achieving 45.0 AP on the COCO dataset with a 3$\times$ training schedule and ResNet-50 FPN, and an inference time of 22 FPS.

As seen in Table 2, all five models (Fast, Faster, Mask, Cascade, and Sparse R-CNNs) offer unique improvements and advancements in object detection. However, each model also faces its own set of challenges and limitations. The Fast R-CNN improved accuracy by using a selective search for object proposals, but it suffered from high computational complexity. The Faster R-CNN improved speed and efficiency by incorporating the RPN, but it had limited scalability and precision. The Mask R-CNN increased accuracy with its binary mask output, but it also had a more complex architecture and limited scalability. The Cascade R-CNN improved accuracy with its series of detection heads, but it had limited scalability and could miss some objects. The Sparse R-CNN improved efficiency with its learnable proposal boxes, but it had limitations in accuracy for smaller objects [38].

Table 2. Comparison of R-CNN models.

Model	Key Features	Improvements	Issues
Fast R-CNN	Utilizes selective search for object proposals.	Improved detection accuracy and reduced computation time.	Reliant on the quality of the object proposals.
Faster R-CNN	Integrates object proposal generation into the network.	Increased speed and accuracy compared to Fast R-CNN.	Limited to fixed-sized object proposals.
Mask R-CNN	Adds object mask output to the bounding box and class predictions.	Improved object detection with the more precise spatial distribution of the object.	Requires additional computation for the mask output.
Cascade R-CNN	Employs a series of detection heads with increasing IoU thresholds.	Improved accuracy by addressing the issue of missed detections.	Increased computation time compared to Faster R-CNN.
Sparse R-CNN	Uses predefined, learnable proposal boxes.	Improved detection accuracy and reduced computation time compared to other R-CNN models.	Limited to the number of proposal boxes defined.

3.2. *YOLO*

The YOLO model, introduced by Joseph Redmon [24], is a state-of-the-art object detection algorithm that offers real-time object detection capabilities. Unlike traditional object detection methods that require multiple scans of an image, YOLO detects objects in a single forward pass through a neural network. In YOLO, the input image is divided into a grid, and a network of bounding boxes is generated for each grid cell to predict the presence and location of objects within the grid. Each bounding box is associated with a probability score indicating the likelihood of detecting a specific object class and the location information encoded in the bounding box. The final predictions are made by selecting bounding boxes with a probability score higher than a predefined threshold. The YOLO framework comprises three main components:

- Backbone extracts high-level features from the input image;

- Neck generates feature pyramids to enable multi-scale object detection;
- Head performs the final detection by applying anchor frames and generating the output vectors with class probabilities, scores, and bounding boxes.

YOLO's single-pass object detection strategy results in faster inference times compared to traditional object detection algorithms such as R-CNN. However, one of the limitations of YOLO is its difficulty in detecting smaller objects in an image. Over the years, several variants of YOLO have been proposed, including YOLOv3 [39], YOLOv4 [40], YOLOv5 [41], YOLOv6 [42], and YOLOv7 [43], which differ in the backbone, neck, head, and loss function used, as shown in Table 3. These variants offer improved performance and accuracy, making YOLO a powerful tool for object detection in various applications.

Table 3. Comparison of inner structure of YOLO architectures.

Model	Type	Backbone	Neck	Head	Loss Function	Framework
YOLOv3	Fully convolution	Darknet-53	FPN	YOLOv3 head	Binary cross entropy	Darknet
YOLOv4	Fully convolution	CSPDarknet53	SPP and PANet	YOLOv3 head	Binary cross entropy	Darknet
YOLOv5	Fully convolution	CSPDarknet53	PANet	YOLOv3 head	Binary cross entropy and Logits loss	Pytorch
YOLOv6	Fully convolution	EfficientRep	Rep-PANet	Decoupled head	Varifocal Loss and Distribution Focal Loss	Pytorch
YOLOv7	Fully convolution	E-ELAN	FPN and PANet	Multiple heads	Binary cross entropy with Focal loss	Pytorch

3.2.1. YOLOv3

YOLOv3 [39] has a total of 106-layer architecture, including 53 convolutional layers. This model has nearly tripled the number of CNN layers compared to YOLOv2. This allows for the detection of objects on three different scales, which is a major improvement over the previous version. One of the key features of YOLOv3 is its ability to resample or down sample, the incoming image, which determines the different scales at which objects can be detected. This resampling leads to the detection of 10-times more frames than was possible with YOLOv2, sacrificing some detection speed for improved accuracy. Additionally, YOLOv3 includes skipping connections and up sampling, which further improves its ability to detect small objects. Another notable difference between YOLOv3 and its predecessor is the method used for class prediction. While earlier versions used a Softmax function, YOLOv3 employs independent logistic classifiers. This allows a single object to belong to multiple classes, making the system more flexible and accurate.

3.2.2. YOLOv4

YOLOv4 [40] brings several significant improvements over its predecessor, YOLOv3. The most notable change is the integration of the Cross-Stage Partial Network (CSP-Net) into the Darknet architecture, creating a new backbone feature extractor known as CSP-Darknet53. The convolutional architecture used in YOLOv4 is based on a modified version of the DenseNet model, which has several advantages over traditional models. These advantages include improved gradient stability, facilitated back-propagation, reduced computational requirements, and enhanced learning. The Neck layer in YOLOv4 is a combination of SPP [44] and Path Aggregation Network (PANet) [45], which enhances the detection process by increasing the receptive field and filtering out essential features. This results in improved accuracy and reduces computational time. YOLOv4 also utilizes "Bag of Freebies" and "Bag of Specialties," which have been proven to significantly improve the performance of the algorithm. The “Bag of Specialties” includes features, such as the Mish activation function and modified PANet, which are not available in previous versions.

3.2.3. YOLOv5

YOLOv5 [41] offers significant improvements over its predecessor, YOLOv4. At the core of YOLOv5 is the combination of the CSP-Darknet53 network and the Focus layer, which form an efficient and highly accurate backbone. This backbone solves the problem of repeating gradient information in large networks, resulting in faster inference times, improved accuracy, and reduced model size. One of the key changes in YOLOv5 is the replacement of the BottleneckCSP block with the C3 block, which features three convolutional layers and enhances the information flow. PANet acts as a bottleneck, improving the precision of object localization, especially for tiny and large objects. The SPP layer helps overcome the limitation of fixed network size, and the use of the SiLU activation function instead of the Mish activation function adds to the efficiency of the system. Another innovative feature of YOLOv5 is its use of mosaic augmentation, where the network is trained on images created from four training photos. This allows for smaller training batch sizes and enables the use of GPUs with lower memory. YOLOv5 also offers a range of five different architecture options, including YOLOv5 Nano, Small, Medium, Large, and XLarge. Each architecture is designed to meet different size, speed, and precision requirements.

3.2.4. YOLOv6

YOLOv6 [42] has been designed to ensure optimal performance and to accommodate the hardware’s capabilities, specifically the CPU power and memory bandwidth. To accomplish this, the backbone and neck architecture of YOLOv6 were re-designed utilizing the Rep-PANet and EfficientRep technology, making it faster and more efficient. One of the major improvements in YOLOv6 is its decoupled head architecture, which allows it to detect objects at double the speed of the previous version, YOLOv5, without sacrificing accuracy. Eliminating shared features of box categorization and regression allows YOLOv6 to be more precise in object detection, resulting in fewer false-positive and false-negative results. Additionally, YOLOv6 adopts an anchorless paradigm, which resulted in improved detection accuracy. The SimOTA label assignment procedure and SIoU Bounding Box regression loss further enhance the accuracy of the object detection system. At present, YOLOv6 is available in three different sizes, Nano, Tiny, and Small, to cater to a wide range of applications, from small edge devices to high-performance systems.

3.2.5. YOLOv7

YOLOv7 [43] is the latest advancement in object detection algorithms. YOLOv7 brings several important improvements that make it a more powerful tool for object detection. One of the major differences between YOLOv7 and YOLOv6 is the backbone network. YOLOv7 uses an Extended Efficient Layer Aggregation Network (E-ELAN) instead of the EfficientRep used in YOLOv6. The E-ELAN makes use of the cardinality of expand, shuffle, and merge operations to continually enhance the network’s learning potential while preserving the gradient path. This leads to improved accuracy and speed in object detection tasks. Another improvement in YOLOv7 is the use of the FPN and PANet in the neck. This results in an FPN-PANet structure that combines the best of both networks. The primary head classifies objects, while the secondary heads train in the middle layers, leading to improved accuracy and faster detection times.

In addition to the improved network architecture, YOLOv7 also utilizes Neural Architecture Search (NAS) instruments to optimize the scaling of models for deployment. This NAS-based approach increases scalability and robustness, as the NAS algorithm iteratively identifies the optimal scaling factors based on resolution, width, depth, and stage (number of feature pyramids). Finally, YOLOv7 also employs module-level re-parameterization, where different segments of the model are optimized using exclusive strategies. The gradient flow propagation channels determine which segments of the model require recalibration, leading to improved robustness and accuracy.

4. Methodology

In this study, an overview of current visual data-based pothole detection is performed. According to the findings obtained, the R-CNN and YOLO models were widely used architectures within the related literature.

As for the evaluation part, selected models from R-CNN and YOLO families were tested for their detection performance on pothole dataset. It was investigated to what extent negative light and weather impacts the accuracy of object detectors. The research objective was formulated as follows: does the current state of the art have adequate tools to effectively handle deviations from clear conditions?

4.1. Experimental Setting

The proposed models in this study were tested under laboratory conditions, and the results were analysed both quantitatively and qualitatively. The operating system utilized was Windows 10, and the tools and environment used for deep learning model training included Anaconda3, CUDA v11.3 with a model acceleration training function, cuDNN8, and PyTorch 1.11.0 as the training framework. The hardware used was a high-performance CPU: Intel Core i9 12900HX Alder Lake and a powerful GPU: NVIDIA GeForce RTX 3090 Ti 24GB.

The transfer learning technique, which plays a crucial role in deep learning, was employed in this study. The technique focuses on leveraging information from a previously trained network (pre-trained weights) and applying it to related problems. The ImageNet and COCO dataset, which are widely used for object detection, were employed in computing pre-trained weights that were applied for model training in this study. The R-CNN models used in this study are available in [46]. Each R-CNN model utilizes ResNet-50 backbone for feature extraction. Furthermore, publicly available implementations of the YOLO architecture were utilized, and larger models were selected to match the size of the R-CNN network. A visual representation of the workflow of the detection process is given in Figure 2. It should be noted that conversion of annotation to COCO format was performed for the training of R-CNN models.

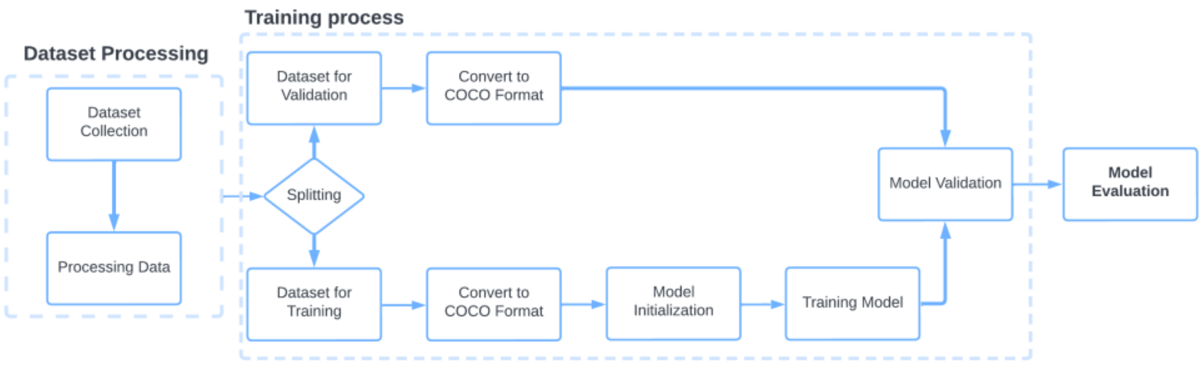

Figure 2. Diagram of the model training process.

Before training, each detector required the configuration of a set of hyperparameters. The default values were used as a starting point and several models were trained by changing one or more of the main hyperparameters. The models with converged loss values and optimal performance on the validation set were selected for this research. The training process parameters of the RCNN and YOLO models are presented in Table 4. The SGD momentum optimizer was utilized for all models. The Early Stopping function automatically halted the training process when the validation loss did not improve for five consecutive epochs.

Table 4. Training parameters for R-CNN and YOLO models.

RCNN Models				YOLO Models			
Parameter	**Value**	**Parameter**	**Value**	**Parameter**	**Value**	**Parameter**	**Value**
Learning Rate	0.0025	Weight Decay	0.0001	Learning Rate	0.001	Weight Decay	0.0005
Batch Size	8	Momentum	0.8	Batch Size	16	Momentum	0.9
Image Size	1080 × 1080	Epochs	300	Image Size	1080 × 1080	Epochs	300

4.2. Dataset

In the field of pothole detection, it is of the utmost importance to use datasets that closely simulate real-world scenarios. This typically involves a camera mounted on a moving vehicle that captures images while on the road in adverse weather conditions. Unfortunately, the existing datasets for pothole recognition fall short of replicating these conditions. To tackle this issue, we created a database that is accessible in [47].

Our image data collection showcases a wide range of potholes, including those that pose significant road hazards. The images with a resolution of 1920 × 1080 were captured from a single stretch of road located in an industrial area of a city, known for its inadequate road conditions, over a period of three months (May, June, and July). This timeline allows for the capture of the evolving surroundings, such as the presence of pedestrians, passing or stationary vehicles, and changes in road conditions. Each image is stamped with an exact date and time of capture.

The dataset also includes images of manhole covers, which can pose a challenge to the model as they can be easily mistaken for potholes due to their circular shape. This step helps to improve the generalization capability of the computer vision model and reduce the likelihood of false positives. The annotated collection of 1052 photos was captured during clear weather conditions, but the dataset also covers four unfavourable weather conditions: rain, sunset, evening, and night. These conditions pose a challenge to the detection models; thus, the availability of a realistic dataset is crucial for the development of robust and reliable pothole detection systems. Figure 3 provides a visual representation of the real-world scenarios, while Table 5 summarizes the statistics of the dataset. In addition to the number of images in each subset, statistics include the number of occurrences of potholes and manhole covers in each of the two categories. For training purposes, the clear dataset was split into training, test, and validation subsets in a 70:15:15 ratio. The data from the remaining adverse conditions were then used for testing only.

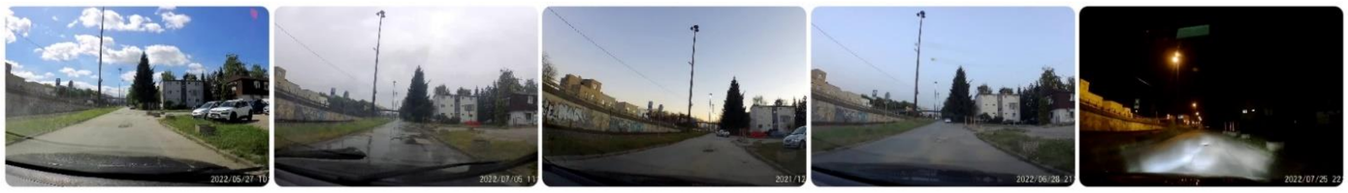

Figure 3. Typical images of potholes on road surfaces (from the left: clear, rain, sunset, evening, night).

Table 5. The pothole dataset description.

Data	Num. of Images	Num. of Instances	Potholes	Manhole Covers
Clear	1052	2128	1896	232
Rain	286	458	383	75
Sunset	201	404	364	40
Evening	250	339	286	53
Night	310	262	220	42

4.3. Dataset Augmentation

Data augmentation is a crucial technique in deep learning to mitigate the impact of limited dataset size and prevent overfitting. By artificially increasing the size of the dataset, data augmentation helps models to learn and generalize better, leading to improved accuracy. In the case of YOLO and R-CNN, various data augmentation techniques were used to improve the performance of the models. In YOLO, the unique technique of mosaic augmentation was employed. This involves combining multiple randomly cropped images into a grid, creating a more diverse set of data for the model to learn from. The following parameters were used for data augmentation in YOLO training: scale factor of 0.5, shear of 0.5, up and down flip of 0.2, left–right flip of 0.5, mosaic of 1, and translation of 0.1. Similarly, R-CNN performance was enhanced through the use of augmentation techniques, including flipping and translation. The parameters used were a flip factor of 0.5, a vertical flip factor of 0.008, and a horizontal flip factor of 0.5. These augmentations helped R-CNN and YOLO to avoid overfitting and generalize better to new data, leading to improved accuracy.

4.4. Evaluation Metrics

The accuracy of an object detector can be evaluated according to a variety of parameters. The most typical assessment metrics include precision, recall, and Average Precision (AP) or mAP. The AP metric is intended to offer a reliable and consistent assessment of the categorization and detection processes. In addition, the frame rate is a key indicator of the speed of the object detector; therefore, it is important to pay attention to both metrics. To describe the mAP metric, it is important to define the following terms first:

- Precision is the ratio of successfully recognized occurrences, denoted as True Positives (TPs), to all positively detected instances (TP + False Positivess (FP)), as shown in Figure 4. Recall is defined as the ratio of correctly recognized instances to all tested instances (TPs + False negatives (FNs)).
- The IoU algorithm is used to calculate the amount of overlap that exists between the anticipated and ground truth bounding boxes. Then, the detection of TP is referred to be a match between bounding boxes that is greater than a particular threshold. The occurrence of FP takes place when the detection level falls below a predetermined threshold. The fact that the proper detection was not made is indicated by the FN instance.

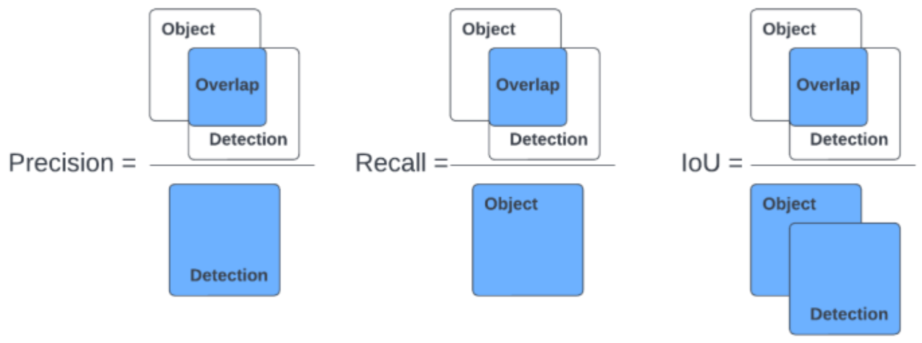

Figure 4. Visual comparison of evaluation metrics.

The accuracy of the prediction can be determined by using the precision/recall (PR) curve and the Area Under Curve (AUC). The AP measure that was decided upon for The PASCAL Visual Object Classes Challenge 2010 [48] is calculated from the PR curve through the process of interpolating the precision at eleven different recall levels [0, 0.1, . . . , 1] (Equation (1)). $\rho_{\text{interp}}(\mathrm{r})$ specifies the precision at each recall level, and it is interpolated by taking the greatest precision measured for which the corresponding recall is greater than r.

$$\mathrm{AP} = \frac{1}{11} \sum_{\mathrm{r} \in \{0, 0.1, \ldots, 1\}} \rho_{\text{interp}}(\mathrm{r}) \tag{1}$$

The notation of mAP is an abbreviation for "mean absolute precision," which is determined by taking the average of all n categories (Equation (2)). The mAP is typically evaluated with different values of IoU threshold, e.g., mAP@.5 means the mean of AP calculated for each class with an IoU threshold > 0.5; if not given, mAP is calculated across all IoU threshold values. The mAP@ [0.5:0.95] is a popular measure that provides the most information about the quality of object identification and it was used as part of the COCO Detection Challenge [49]. It refers to the mAP calculated across an IoU threshold between 0.5 and 0.95. A high score for this metric indicates that the model can identify and correctly classify visual objects.

$$\mathrm{mAP} = \frac{1}{n} \sum_{i=1}^{n} \mathrm{AP}_i \tag{2}$$

5. Results and Discussion

The comparison between R-CNN and YOLO models has been a topic of interest in the field of object detection for a long time. Both families of models have been widely used for object detection tasks, but each has its strengths and weaknesses. In this research, we evaluate the performance of R-CNN and YOLO models on a database that contains different weather conditions (clear, rainy, sunset, evening, and night).

5.1. Performance of R-CNN

According to the evaluation results shown in Table 6, R-CNN-based models' performance in different weather conditions shows a clear correlation between the complexity of the weather conditions and the decrease in precision, recall, and mAP scores. In clear weather, all models performed relatively well, with precision and recall scores ranging from 0.475 to 0.711, mAP@.5 scores ranging from 0.672 to 0.746, and mAP@ [0.5:0.95] scores ranging from 0.269 to 0.338.

Table 6. Results of R-CNN models in different weather conditions.

	Data Subset	**Precision**	**Recall**	**mAP@.5**	**mAP@ [0.5:0.95]**
R-CNN	Clear	64.2%	45.5%	67.2%	26.9%
	Rain	49.4%	42.4%	41.1%	12.7%
	Sunset	51.2%	45.9%	46.4%	15.8%
	Evening	47.5%	41.6%	38.8%	10.4%
	Night	31.6%	24.3%	29.7%	8.1%
Fast R-CNN	Clear	67.9%	48.4%	71.9% + 4.7	29.2% + 2.3
	Rain	51.3%	44.0%	44.0% + 2.9	14.8% + 2.1
	Sunset	53.2%	46.7%	48.1% + 1.7	17.5% + 1.7
	Evening	48.8%	43.3%	39.4% + 0.6	12.8% + 2.4
	Night	32.7%	31.6%	31.0% + 1.3	9.2% + 1.1
Faster R-CNN	Clear	69.2%	50.3%	73.6% + 1.7	32.1% + 2.9
	Rain	53.4%	45.2%	45.5% + 1.5	15.9% + 1.1
	Sunset	55.7%	47.4%	50.2% + 2.1	18.7% + 1.2
	Evening	49.6%	44.5%	41.3% + 1.9	13.6% + 0.8
	Night	33.9%	32.7%	31.8% + 0.8	10.2% + 1.0

Table 6. *Cont.*

	Data Subset	**Precision**	**Recall**	**mAP@.5**	**mAP@ [0.5:0.95]**
Mask R-CNN	Clear	64.1%	53.4%	70.3% − 3.3	28.3% − 3.8
	Rain	55.2%	41.7%	48.3% + 2.8	16.2% + 0.3
	Sunset	51.6%	44.6%	43.7% − 6.5	18.6% − 0.1
	Evening	57.5%	47.2%	53.8% + 12.5	22.3% + 8.7
	Night	32.4%	32.7%	28.6% − 3.2	8.9% − 1.3
Cascade R-CNN	Clear	71.1%	55.2%	73.7% + 0.1	33.8% + 1.7
	Rain	56.4%	46.6%	46.1% − 2.2	16.6% + 0.4
	Sunset	57.0%	48.1%	52.4% + 2.2	19.5% + 0.8
	Evening	51.5%	45.8%	47.5% − 6.3	14.4% − 7.9
	Night	34.8%	33.5%	33.2% + 1.4	10.6% + 0.4
Sparse R-CNN	Clear	69.2%	54.2%	74.6% + 0.9	32.1% − 1.7
	Rain	58.7%	47.7%	48.9% + 0.6	16.6% + 0.0
	Sunset	56.9%	48.3%	49.5% − 2.9	19.1% − 0.4
	Evening	59.0%	48.0%	54.1% + 0.3	22.9% + 0.6
	Night	33.6%	34.7%	29.8% − 3.4	9.4% − 1.2

The Faster R-CNN, Cascade R-CNN, and Sparse R-CNN models are leading in terms of mAP success rates, with the Faster R-CNN and Sparse R-CNN having the highest precision score of 0.692 and the Cascade R-CNN having the highest recall score of 0.552. They also showed the largest performance improvement compared to the standard R-CNN model. Faster R-CNN improved by 5% in precision and 4.8% in recall, Cascade R-CNN improved by 6.9% in precision and 9.7% in recall, and Sparse R-CNN improved by 5% in precision and 8.7% in recall.

Under more challenging conditions, such as rain and changing light, due to the late hours of the day, R-CNN models saw a significant drop in performance. With the improvement in models and their detection accuracy under clear conditions, the gap in the performance between clear and worsened conditions may also increase. However, there are some exceptions, especially with the newer architectures, such as Cascade and Sparse R-CNN. Each model responds differently to adverse road conditions, but one of the common phenomena was a significant decrease in the accuracy of detection under night conditions. When compared to models' performance under clear conditions, there was a decrease recorded, on average, by 26%, 23.5%, 20.3%, and 41.2% in mAP@.5 for the rain, sunset, evening, and night subsets, respectively.

The continuous change in the mAP accuracy throughout the different versions of R-CNN models is marked in Table 6, while the accuracy of the basic R-CNN model is considered as the baseline starting point. The difference in accuracy is calculated considering the current highest value achieved for individual data subsets.

A comparison of the overall performance of the models, when the mean value of accuracy across all conditions is calculated, is shown in Table 7. We can see that the various object detection models function differently. To better understand the strengths and weaknesses of each model, we will compare them in several ways, including model size, number of parameters, inference time, and performance.

Table 7. Performance of R-CNN models under multiple indicators.

Model	Mean_P	Mean_R	Mean_mAP@.5	Number of Parameters	Model Size	Epoch	Inference Time
R-CNN	48.7%	39.9%	44.6%	34 M	149.4 MB	194	189.8 ms
Fast R-CNN	50.8%	42.8%	46.8%	42.5 M	223.7 MB	233	178.5 ms
Faster R-CNN	52.3%	44.0%	48.4%	53.2 M	271.2 MB	227	175.2 ms
Mask R-CNN	52.1%	43.9%	48.9%	64.1 M	370.8 MB	246	141.6 ms
Cascade R-CNN	54.2%	45.8%	50.5%	107 M	537.3 MB	274	214.1 ms
Sparse R-CNN	55.4%	46.6%	51.4%	77.8 M	415.4 MB	251	146.4 ms

- Model size and inference time: The model size is an important factor to consider when deploying the model in real-world scenarios, as larger models require more memory and computational resources. The inference time, on the other hand, is important for real-time applications, where quick responses are essential. With the smallest model size of 149 MB, the basic R-CNN model provides the inference of 189.8 ms and the least accurate detection. R-CNN successors, Fast and Faster R-CNN, come with an increase in model size as well as in detection accuracy and an improvement in inference speed. Mask and Sparse R-CNN, whose size is more than double that of R-CNN, can achieve both short inference time and higher detection performance. While a multi-stage object detector, Cascade R-CNN, can provide more accurate results, the processing time of the inference is still high.
- Parameters: The number of parameters is another critical factor in determining the model's performance. A model with more parameters has the potential to learn more complex relationships in the data, which can lead to better performance. However, having more parameters also means that the model is more prone to overfitting and requires more computational resources during both training and inference. In this comparison, Cascade R-CNN has the largest number of parameters, with 107 million, followed by Sparse R-CNN with 77.8 million and Mask R-CNN with 64.1 million parameters. On the other hand, Fast and Faster R-CNN are some of the smallest models, with 42.5 million and 53.2 million parameters, respectively.
- Performance: To evaluate the ability of models to cope with all adverse conditions, the mean detection performance over all conditions is considered in this comparison. Sparse R-CNN has the highest overall accuracy, with a mean precision of 55.4%, a mean recall of 46.6%, and the highest mean mAP@.5 value of 51.4%. The differences in performance can be attributed to the architectures of the models and the methods they use to extract features from the input image.
- Epoch: The number of epochs indicates the number of times the model has been trained on the dataset. The number of epochs needed for the model to be trained varies due to the application of the early stopping function. Apart from R-CNN, Fast R-CNN and Faster R-CNN had the smallest number of epochs, with 233 and 227, respectively. Meanwhile, the largest model Cascade R-CNN also has the largest number of epochs, with 274.

Overall, the results in Table 7 suggest that the different models have their strengths and weaknesses, and the choice of model will depend on the specific requirements of the task at hand. For instance, if accuracy is a priority, Cascade R-CNN and Sparse R-CNN are suitable models, while if inference speed is a concern, Mask R-CNN with the lowest inference time is more suitable. Ultimately, Sparse R-CNN provides both the highest accuracy and a low inference time. These results could be because of the differences in the architectures and optimization techniques used in each of these models. The Cascade R-CNN model has a multi-stage architecture, which enables it to refine its predictions at each stage, resulting in improved performance. On the other hand, the Sparse R-CNN model uses a sparse proposal generation technique, which helps it to be more efficient and faster, leading to better performance.

5.2. Performance of YOLO

Table 8 shows a comparison of the performance of different YOLO object detection models. As we can see, the latest models (YOLOv6 and YOLOv7) perform better than the older ones (YOLOv3 and YOLOv4) in terms of precision, recall, and mAP across different weather conditions, with improvements observed in the worsened lighting conditions.

Table 8. Results of YOLO models on different weather conditions.

	Data Subset	**Precision**	**Recall**	**mAP@.5**	**mAP@ [0.5:0.95]**
YOLOv3	Clear	77.7%	75.2%	77.1%	33.0%
	Rain	61.3%	51.9%	50.5%	19.9%
	Sunset	69.4%	49.6%	52.9%	19.4%
	Evening	74.2%	48.3%	47.4%	18.2%
	Night	36.0%	15.7%	17.5%	6.2%
YOLOv4	Clear	79.3%	76.6%	78.7% + 1.6	33.7% + 0.7
	Rain	62.6%	53.0%	51.6% + 1.1	20.3% + 0.4
	Sunset	70.9%	50.6%	54.0% + 1.1	19.8% + 0.4
	Evening	75.8%	49.3%	48.4% + 1.0	18.6% + 0.4
	Night	36.8%	16.0%	17.9% + 0.4	6.3% + 0.1
YOLOv4-csp	Clear	81.1%	80.4%	80.4% + 1.7	34.4% + 0.7
	Rain	63.9%	54.1%	52.7% + 1.1	20.8% + 0.5
	Sunset	72.4%	51.7%	55.2% + 1.2	20.2% + 0.4
	Evening	77.4%	50.4%	49.4% + 1.0	19.0% + 0.4
	Night	37.6%	16.4%	18.3% + 0.4	6.5% + 0.2
YOLOv5m	Clear	84.0%	81.2%	83.3% + 2.9	35.7% + 1.3
	Rain	66.3%	56.1%	54.6% + 1.9	21.5% + 0.7
	Sunset	75.0%	53.6%	57.2% + 2.0	21.0% + 0.8
	Evening	80.2%	52.2%	51.2% + 1.8	19.7% + 0.7
	Night	38.9%	17.0%	18.9% + 0.6	6.7% + 0.2
YOLOv5l	Clear	86.1%	85.4%	85.4% + 2.1	36.6% + 0.9
	Rain	67.9%	57.5%	55.9% + 1.2	22.0% + 0.5
	Sunset	76.9%	54.9%	58.6% + 1.4	21.5% + 0.5
	Evening	82.2%	53.5%	52.5% + 1.3	20.2% + 0.5
	Night	39.9%	17.4%	19.4% + 0.5	6.9% + 0.2
YOLOv6m	Clear	87.9%	84.6%	87.3% + 1.9	37.4% + 0.8
	Rain	69.4%	58.7%	57.2% + 1.3	22.5% + 0.5
	Sunset	78.6%	56.1%	59.6% + 1.0	22.0% + 0.5
	Evening	84.0%	54.7%	53.7% + 1.2	20.6% + 0.4
	Night	40.7%	17.8%	19.8% + 0.4	7.0% + 0.1
YOLOv6l	Clear	88.2%	86.1%	87.8% + 0.5	37.6% + 0.2
	Rain	69.5%	59.1%	57.4% + 0.2	22.7% + 0.2
	Sunset	78.8%	56.4%	59.9% + 0.3	21.0% − 1.0
	Evening	84.4%	55.0%	54.1% + 0.4	20.8% + 0.2
	Night	41.1%	18.6%	19.9% + 0.1	7.0% + 0.0

Table 8. *Cont.*

	Data Subset	Precision	Recall	mAP@.5	mAP@ [0.5:0.95]
YOLOv7	Clear	88.6%	88.0%	88.4% + 0.6	40.3% + 2.7
	Rain	70.6%	60.2%	58.1% + 0.7	24.9% + 2.2
	Sunset	79.1%	57.7%	61.5% + 1.6	22.8% + 0.8
	Evening	84.9%	56.1%	54.8% + 0.7	21.5% + 0.7
	Night	42.8%	19.5%	20.4% + 0.5	7.5% + 0.5

YOLOv5 and YOLOv6 models demonstrated improvements over their predecessors, YOLOv3 and YOLOv4. On average, YOLOv5m shows a 4.6% increase in precision, recall, and mAP@.5 compared to YOLOv4. On top of that, YOLOv6 saw about a 3.9% increase in precision, 3.4% in recall, and 4% in mAP@.5 compared to YOLOv5. YOLOv7 achieved the highest results on the evaluated pothole dataset. On average, the precision, recall, and mAP@.5 of 88.3% were reached, and the highest mAP@ [0.5:0.95] of 40.3% was also achieved. When comparing the baseline YOLOv3 model to the latest YOLOv7 model, we can see a substantial improvement in the overall performance. YOLOv7 showed a 10.9%, 12.3%, 11.3%, and 7.3% improvement in precision, recall, mAP@.5, and mAP@ [0.5:0.95] score, respectively, compared to YOLOv3 under clear weather conditions.

Adverse weather and reduced light have a natural impact on camera sensing and subsequent object detection performance. As in the previous case of R-CNN models, the performance of YOLO models degraded when subjected to more difficult visual conditions. Despite the detection accuracy being continuously improved in clear settings, the performance difference between clear and deteriorated conditions may also widen. The mAP@.5 for rain, sunset, evening, and night subsets decreased, on average, by 28.8%, 26.2%, 32.11%, and 64.5% in comparison to clear conditions.

The continuous change in the mAP accuracy throughout the different versions of YOLO models is marked in Table 8. The accuracy of the YOLOv3 model is considered the baseline starting point. The difference in accuracy is calculated considering the current highest value achieved for individual data subsets.

As can be seen from Table 8, each new version of YOLO comes with additional improvements in the mAP@.5 measure for every data subset. However, YOLO's detection performance rapidly declines at night. Unlike RCNN models reaching up to 30% of mAP@.5, the detection accuracy of YOLO models does not hit above the 20% level of mAP@.5.

A comparison of the overall performance of the models, when the mean value of accuracy across all conditions is calculated, is shown in Table 9. The accuracy of the pothole detection task increases with the newer versions of YOLO architecture. In addition, we compare individual YOLO models from several perspectives, including model size, number of parameters, inference time, and performance.

Table 9. Performance of YOLO models under multiple indicators.

Model	Mean_P [%]	Mean_R [%]	Mean_mAP@.5	Parameter	Size	Epoch	InferenceTime
YOLOv3	63.7	48.1	49.1	63.0 M	125.8 MB	192	95.2 ms
YOLOv4	65.1	49.1	50.1	63.9 M	246 MB	183	92.6 ms
YOLOv4-csp	66.4	50.6	51.2	61.1 M	202 MB	197	91.0 ms
YOLOv5m	68.8	52.0	53.0	22.0 M	42.4 MB	161	79.8 ms
YOLOv5l	70.6	53.7	54.3	47.9 M	91.9 MB	179	83.9 ms
YOLOv6m	72.1	54.4	54.8	34.9 M	72.6 MB	142	81.7 ms
YOLOv6l	72.4	55.0	55.0	59.6 M	114.0 MB	158	84.3 ms
YOLOv7	73.2	56.3	55.6	36.9 M	35.5 MB	155	78.3 ms

- Model size and inference time: YOLOv7 model has the smallest size of 35.5 MB, which is nearly 1/7 of the largest YOLOv4 model and over 1/3 of the YOLOv3 model, with sizes of 246 MB and 125.8 MB, respectively. Moreover, YOLOv7 achieved the fastest inference time of 78.3 ms, with a 12.9% improvement compared to the slowest inference time of 95.2 ms of YOLOv3. The YOLOv5m and YOLOv7 models have shown particularly impressive results in this aspect, with 42.4 MB and 35.5 MB in size, respectively, and 79.8 ms and 78.3 ms in inference time.
- Parameters: YOLOv5m model has the smallest number of parameters of 22.0 M, with a 65.5% reduction compared to YOLOv4, which is the model with the largest number of parameters (63.9 M). This is followed by YOLOv7 with a 42.2% parameter reduction compared to YOLOv4. For further "compression", the Tiny or S size type of models can be utilized. However, this will often come at the cost of reduced precision.
- Performance: By comparing models' mean detection performance across all different data subsets, we can see how well they function under adverse circumstances. Overall, YOLOv7 performed best, with mean precision of 73.2%, mean recall of 56.3%, and mean mAP@.5 of 55.6%. Moreover, YOLOv7 shows the best inference time compared to other YOLO models. The YOLOv6l and YOLOv7 models have shown particularly good overall results with 72.4% and 73.2% mean precision and 55.0% and 56.3% mean recall, respectively.
- Epoch: YOLOv6m model has the shortest training time at 142 epochs, with a 28% reduction compared to the longest training time of 197 epochs of YOLOv4-csp. The YOLOv7 shows a 21% reduction in the number of epochs compared to YOLOv4-csp.

The YOLOv7 model has the best performance, the smallest size, and the fastest inference time. YOLOv7 is suitable for applications with lower performance requirements or computational constraints. This model has shown significant improvements in all aspects compared to other models, making it a strong contender for various object detection applications. It is important to note that the YOLOv5m and YOLOv6m models also showed impressive results, particularly in the areas of accuracy, model size, and inference time.

5.3. Comparison of R-CNN and YOLO Results

Based on the performance metrics presented in Sections 5.1 and 5.2, we can draw several conclusions about the relative strengths and weaknesses of R-CNN and YOLO for the task of pothole detection. The results demonstrate that YOLO models outperform R-CNN models in terms of accuracy and speed of inference. Figure 5 shows the detection accuracy of the best-performing YOLO and R-CNN architectures. As can be seen, the models from the R-CNN group achieved higher results (up to 30% mAP@.5) on the night data subset, which presents worse possible conditions. In conclusion, the overall accuracy represented by Mean_mAP@.5 shows that R-CNN models are competitive regarding the detection task under adverse visual conditions.

A visual comparison of models' accuracy versus the number of parameters and model size is shown in Figure 6. YOLOv7 is the most efficient model, with a Mean_mAP@.5 of 55.6%. The best-performing R-CNN model, Sparse R-CNN, achieved a Mean_mAP@.5 of 51.4%. YOLOv7 has a smaller model size of 35.5 MB and requires only 36.9 million parameters, which is approximately a 91.5% and 52.6% reduction compared to the Sparse R-CNN model, which has a size of 415.4 MB and 77.8 million parameters. This makes YOLOv7 highly effective for real-time object detection in applications such as pothole detection.

When comparing YOLOv7 to R-CNN models in terms of detection speed, the percentage improvement is substantial. The YOLOv7 model achieved an inference time of 78.3 ms, which is approximately a 46.1% reduction compared to the Sparse R-CNN model, with an inference time of 146.4 ms. Additionally, with each new version of the YOLO model, performance continues to improve.

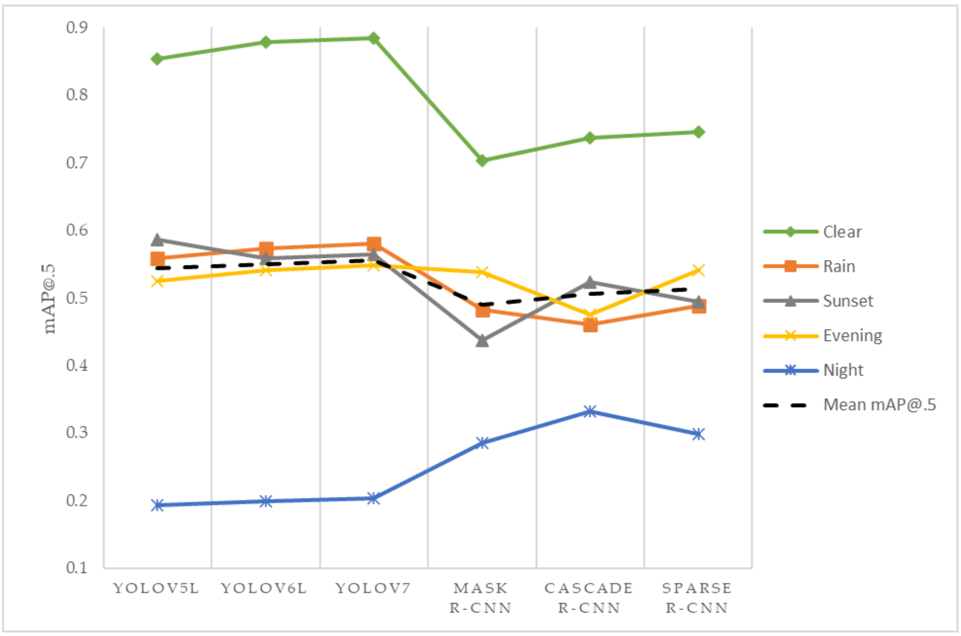

Figure 5. The accuracy of selected YOLO and R-CNN architectures in terms of pothole detection under diverse visual conditions.

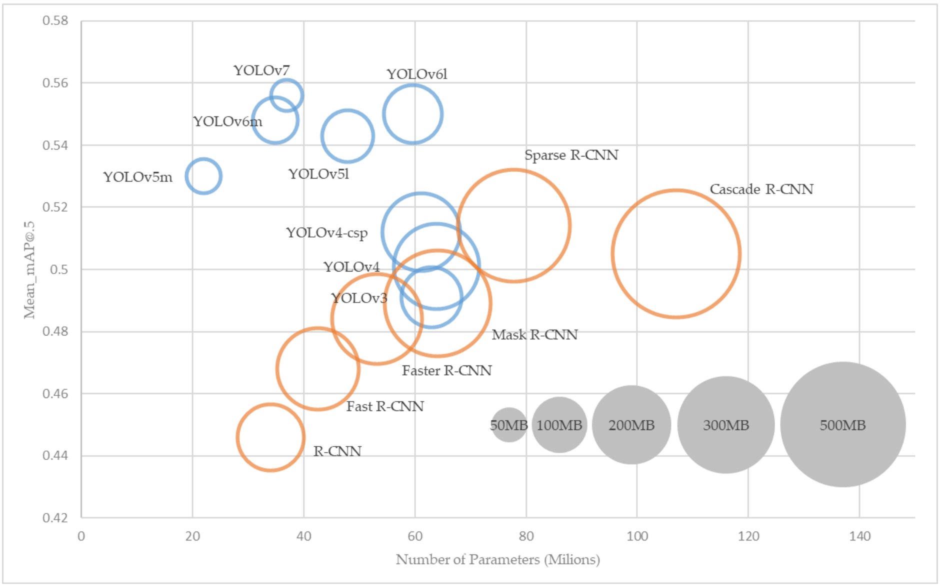

Figure 6. Comparison of the prediction accuracy versus the number of parameters and storage requirements.

Interestingly, the YOLOv5m model has a higher Mean_mAP@.5 (53.0%) than all the R-CNN models, while also having a smaller number of parameters and model size than most R-CNN as well as some YOLO models. These results demonstrate the efficiency and effectiveness of the YOLO architecture in the pothole detection task, particularly with its latest versions of YOLOv5, YOLOv6, and YOLOv7.

Despite R-CNN-based models having significantly more parameters, model size, and requiring longer inference time, they can extract useful features that contribute to the improvement in pothole detection, especially under the worst visibility at night.

The findings obtained throughout this work have practical implications for pothole detection applications, as YOLOv7 can accurately detect potholes in real time with a low computational cost, making it suitable for deployment on resource-constrained devices.

This can potentially lead to more efficient road maintenance and improved road safety. It is important to note that the performance of any object detection algorithm is highly dependent on the quality and size of the dataset used for training, as well as the specific implementation details.

5.4. Mitigating Adverse Visual Conditions

Computer vision systems are challenged by low-contrast images, shadows, or adverse weather. Several research works have been conducted to mitigate the effects of adverse visual conditions on camera sensing and subsequent object detection.

Deep learning has shown impressive performance in low-light image enhancement. Techniques, such as multiscale [50,51] and attention feature maps [52], have been successfully utilized. An end-to-end attention-based multi-branch CNN was developed in [53]. It performs denoising and low-light enhancement simultaneously to deal effectively with colour distortion and noise that are also occurring within dark images. Consisting of four subnetworks, the system creates an under-exposed attention map to handle the under exposed image parts and then the noise map is derived. Both image representations are further used in a multi-branched subnetwork for the image enhancement task. A final separate CNN is utilized for contrast, exposure, and colour improvement. The proposed model outperformed other state-of-the-art methods in terms of enhancement rate, with testing time reported as 0.05 s for the lightweight model version and 0.48 s for the model tested on the SID dataset. Apart from low-light image enhancement, rainy images can also be effectively derained [54,55] or defogged [56,57]. The high-quality image enhancement techniques can be used as a pre-processing step to improve object/pothole detection. However, the computational constraints and overall inference speed should be considered.

Another approach is to perform enhancement and detection jointly in an end-to-end manner. According to [58], a learnable pre-processing module for low-light image enhancement may decrease the accuracy of detection in some cases. The authors of [58] proposed learnable low-light image enhancement implemented jointly with the detection task utilizing the twin architecture. By using the information at both the original and enhanced features levels, an improvement in face detection was achieved. In [59], an image-adaptive YOLO deals with weather-specific information using a fully differentiable image processing module (DIP) to pre-process high-resolution images for input to YOLOv3. Hyperparameters of the DIP module are learned by separate the CNN predictor networks. The proposed system performed effectively under foggy and low-light scenarios with an inference time increase of 13 ms over the YOLOv3 baseline.

Improving internal modules of existing object detection architecture may increase model robustness and allow for better regulation of the model speed. A Trans-Decoupled YOLO [60], which is designed for small object detection in complex environments, integrates transformer modules with a self-attention mechanism into the YOLOv5 backbone for global contextual feature extraction. Additionally, a decoupled lightweight head for both simplified and more accurate detection was proposed. Improvements of 6.4% (mAP@.5) and 6.8% (mAP@ [0.5:0.95]) over YOLOv5 on the TT100K dataset were reported. Convolutional Block Attention Module (CBAM) is another visual attention-based module used to improve not only the small object detection but also to mitigate adverse weather conditions [61,62]. Inspired by CBAM, the authors of [63] proposed a global attention mechanism with 3D permutation in its channel part and increased number of conv layers in the spatial part.

A common approach to improve a model's generalization to adverse conditions is to incorporate images captured under different weather and lighting conditions in the training dataset [9,10]. The available data can also be augmented with synthetically created images or with original images translated into different conditions. For instance, synthetically produced rain data added to training images improved object detection by 21% in [64].

The influence of adverse visual conditions can also be mitigated by the fusion of different sensors. Although accelerometer-based pothole detection is sensitive to vehicle speed, it is particularly useful for detection in low visibility. For more accurate results, video data were combined with an acceleration sensor for vehicle vibration measurements in public crowdsourcing applications in [65]. As in previous cases, the multi-sensor system can be enhanced with an attention mechanism and enabled to adapt effectively to varying adverse weather conditions [66].

6. Conclusions

Computer vision techniques have shown promising results in automating pothole detection, but selecting the best model for deployment can be a challenging task, especially if we consider the detection of potholes under different weather conditions. In this article, we focused on introducing the current state-of-the-art CNNs that are used for pothole detection. The main objective is to compare their performance through experiments and provide information that can be useful for future research in this field. This study evaluated the effectiveness of different computer vision models, including Fast R-CNN, Faster R-CNN, Mask R-CNN, Cascade R-CNN, Sparse R-CNN, and YOLO versions 3 to 7, for the task of pothole detection under adverse visual conditions, such as rain, sunset, evening, and night. The models' performances were compared in terms of detection accuracy under different weather and lighting conditions.

Our experimental results revealed that YOLOv7, followed by YOLOv6l and YOLOv6m, demonstrated the best performance across all weather conditions. YOLOv5l and YOLOv5m also showed good performance, with slight variations in different weather conditions. These results indicate that YOLO architectures may be the most suitable for pothole detection under adverse visual conditions, such as rain, sunset, and evening. However, it is worth noting that R-CNN models, despite their significant computational costs, proved to be the most suitable for night-time detection. Although YOLO architectures perform detection with significant accuracy and speed, R-CNN models may handle the very low-visibility detection more successfully.

The results showed that the performance of the models was negatively affected by lighting conditions, with night data showing the lowest performance. When compared to the clear subset, the mAP@.5 for the night subset decreased, on average, by 41.2% and 64.5% for R-CNN and YOLO models, respectively. These findings highlight the importance of considering different weather and low-light conditions when selecting object detection models. Our study's contributions may provide valuable information for researchers interested in improving pothole detection performance under adverse visual conditions. The proposed study may also contribute to the development of ITS, which aims to improve road safety and reduce the number of accidents caused by potholes.

Future research could focus on more diverse and weather-specific data augmentation techniques using generative networks. These methods could enable the generation of synthetic data that accurately captures the complexities of different weather conditions, thus improving the generalization capability of the models. Moreover, model modification such as self-attention modules (e.g., Transformers, CBAM, GAM) for salient feature extraction could be incorporated to improve the detection of relatively small objects in challenging conditions. Novel multi-scale features could also be implemented to capture objects at different scales and enhance model performance. Furthermore, incorporating additional information into the object detection pipeline, such as semantic segmentation or depth estimation, could help to further improve the accuracy of object detection in challenging weather conditions. Additionally, it would be useful to extend the dataset to other weather conditions (snowfall, hail, fog) to assess the robustness of the models. Finally, an interesting direction for future research would be to implement the models on hardware platforms for real-world testing. This could involve deploying the models on drones or vehicles to evaluate their effectiveness in detecting objects in real time, which would have significant implications for applications, such as autonomous driving and aerial surveillance.

Author Contributions: Conceptualization, M.J., E.L. and B.B.; methodology, K.Z.; software, M.J.; validation, B.B., E.L. and K.Z.; formal analysis, B.B., E.L.; investigation, M.J.; resources, K.Z.; data curation, B.B.; writing—original draft preparation, M.J. and E.L.; writing—review and editing, B.B. and K.Z.; visualization, M.J.; supervision, B.B.; project administration, B.B.; funding acquisition, K.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This publication was realized with the support of Operational Program Integrated Infrastructure 2014–2020 of the project: Innovative Solutions for Propulsion, Power and Safety Components of Transport Vehicles, code ITMS 313011V334, co-financed by the European Regional Development Fund.

EUROPEAN UNION
European Regional Development Fund
OP Integrated Infrastructure 2014 – 2020

MINISTRY
OF TRANSPORT
OF THE SLOVAK REPUBLIC

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data supporting reported results can be found at https://doi.org/10.6084/m9.figshare.21214400.v3 (accessed on 23 March 2023).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Novotny, P.; Janosikova, M. Designating Regional Elements System in a Critical Infrastructure System in the Context of the Czech Republic. *Systems* **2020**, *8*, 13. [CrossRef]
2. Lacinák, M. Resilience of the Smart Transport System—Risks and Aims. *Transp. Res. Procedia* **2021**, *55*, 1635–1640. [CrossRef]
3. American Automobile Association; Pothole Damage Costs U.S. Drivers $3 Billion Annually. 2016. Available online: https://info.oregon.aaa.com/pothole-damage-costs-u-s-drivers-3-billion-annually/ (accessed on 2 May 2023).
4. Drliciak, M.; Celko, J.; Cingel, M. The Economically Active People in the Transport Process. In Proceedings of the IOP Conference Series: Materials Science and Engineering, Žilina, Slovakia, 9–13 September 2019; Volume 661.
5. Siew, E.F.; Ireland-Hay, T.; Stephens, G.T.; Chen, J.J.J.; Taylor, M.P. A Study of the Fundamentals of Pothole Formation. In Proceedings of the Light Metals 2005: Proceedings of the Technical Sessions Presented by the TMS Aluminum Committee at the 134th TMS Annual Meeting, San Francisco, CA, USA, 13–17 February 2005; Volume 1, pp. 763–769.
6. Cingel, M.; Drliciak, M.; Celko, J. Morning Modal Split Model of Economically Active People in Zilina Region. *Proc. Transp. Res. Procedia* **2021**, *55*, 1065–1072. [CrossRef]
7. Heo, D.-H.; Choi, J.-Y.; Kim, S.-B.; Tak, T.-O.; Zhang, S.-P. Image-Based Pothole Detection Using Multi-Scale Feature Network and Risk Assessment. *Electronics* **2023**, *12*, 826. [CrossRef]
8. Zabovsky, M.; Chochlik, M.; Matiasko, K.; Valentikova, E. Dynamic Architecture for Analytical Its Services. *Commun. Sci. Lett. Univ. Zilina* **2010**, *12*, 42–45. [CrossRef]
9. Pena-Caballero, C.; Kim, D.; Gonzalez, A.; Castellanos, O.; Cantu, A.; Ho, J. Real-Time Road Hazard Information System. *Infrastructures* **2020**, *5*, 75. [CrossRef]
10. Ochoa-Ruiz, G.; Angulo-Murillo, A.A.; Ochoa-Zezzatti, A.; Aguilar-Lobo, L.M.; Vega-Fernández, J.A.; Natraj, S. An Asphalt Damage Dataset and Detection System Based on RetinaNet for Road Conditions Assessment. *Appl. Sci.* **2020**, *10*, 3974. [CrossRef]
11. Ahmed, K.R. Smart Pothole Detection Using Deep Learning Based on Dilated Convolution. *Sensors* **2021**, *21*, 8406. [CrossRef]
12. Al-Shaghouri, A.; Alkhatib, R.; Berjaoui, S. Real-Time Pothole Detection Using Deep Learning. *arXiv* **2021**, arXiv:2107.06356.
13. Yu, X.; Marinov, M. A Study on Recent Developments and Issues with Obstacle Detection Systems for Automated Vehicles. *Sustainability* **2020**, *12*, 3281. [CrossRef]
14. Wang, H.-W.; Chen, C.-H.; Cheng, D.-Y.; Lin, C.-H.; Lo, C.-C. A Real-Time Pothole Detection Approach for Intelligent Transportation System. *Math. Probl. Eng.* **2015**, *2015*, 869627. [CrossRef]
15. Harikrishnan, P.M.; Gopi, V.P. Vehicle Vibration Signal Processing for Road Surface Monitoring. *IEEE Sens. J.* **2017**, *17*, 5192–5197. [CrossRef]
16. Wu, C.; Wang, Z.; Hu, S.; Lepine, J.; Na, X.; Ainalis, D.; Stettler, M. An Automated Machine-Learning Approach for Road Pothole Detection Using Smartphone Sensor Data. *Sensors* **2020**, *20*, 5564. [CrossRef] [PubMed]
17. Chang, K.T.; Chang, J.R.; Liu, J.K. Detection of Pavement Distresses Using 3D Laser Scanning Technology. In Proceedings of the International Conference on Computing in Civil Engineering 2005, Cancun, Mexico, 12–15 July 2005; pp. 1–11. [CrossRef]
18. Yu, X.; Salari, E. Pavement Pothole Detection and Severity Measurement Using Laser Imaging. In Proceedings of the IEEE International Conference Electro/Information Technology, Mankato, MN, USA, 15–17 May 2011.
19. Staniek, M. Stereo Vision Method Application to Road Inspection. *Balt. J. Road Bridge Eng.* **2017**, *12*, 38–47. [CrossRef]

20. Hou, Z.; Wang, K.C.P.; Gong, W. Experimentation of 3D Pavement Imaging through Stereovision. In Proceedings of the International Conference on Transportation Engineering 2007, Chengdu, China, 22–24 July 2007; pp. 376–381.
21. Zhang, Z.; Ai, X.; Chan, C.K.; Dahnoun, N. An Efficient Algorithm for Pothole Detection Using Stereo Vision. In Proceedings of the 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Florence, Italy, 4–9 May 2014; pp. 564–568.
22. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 580–587.
23. Girshick, R. Fast R-CNN. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 1440–1448.
24. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26–27 June 2016; pp. 779–788.
25. Salaudeen, H.; Çelebi, E. Pothole Detection Using Image Enhancement GAN and Object Detection Network. *Electronics* **2022**, *11*, 1882. [CrossRef]
26. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.-Y.; Berg, A.C. SSD: Single Shot MultiBox Detector. In Proceedings of the Computer Vision—ECCV 2016, Amsterdam, The Netherlands, 11–14 October 2016; Leibe, B., Matas, J., Sebe, N., Welling, M., Eds.; Springer International Publishing: Cham, Switzerland, 2016; pp. 21–37.
27. Ye, W.; Jiang, W.; Tong, Z.; Yuan, D.; Xiao, J. Convolutional Neural Network for Pothole Detection in Asphalt Pavement. *Road Mater. Pavement Des.* **2021**, *22*, 42–58. [CrossRef]
28. Ma, N.; Fan, J.; Wang, W.; Wu, J.; Jiang, Y.; Xie, L.; Fan, R. Computer Vision for Road Imaging and Pothole Detection: A State-of-the-Art Review of Systems and Algorithms. *Transp. Saf. Environ.* **2022**, *4*, tdac026. [CrossRef]
29. Chen, H.; Yao, M.; Gu, Q. Pothole Detection Using Location-Aware Convolutional Neural Networks. *Int. J. Mach. Learn. Cybern.* **2020**, *11*, 899–911. [CrossRef]
30. Park, S.-S.; Tran, V.-T.; Lee, D.-E. Application of Various YOLO Models for Computer Vision-Based Real-Time Pothole Detection. *Appl. Sci.* **2021**, *11*, 11229. [CrossRef]
31. Maeda, H.; Sekimoto, Y.; Seto, T.; Kashiyama, T.; Omata, H. Road Damage Detection and Classification Using Deep Neural Networks with Smartphone Images. *Comput.-Aided Civ. Infrastruct. Eng.* **2018**, *33*, 1127–1141. [CrossRef]
32. Lin, Y.-C.; Chen, W.-H.; Kuo, C.-H. Implementation of Pavement Defect Detection System on Edge Computing Platform. *Appl. Sci.* **2021**, *11*, 3725. [CrossRef]
33. Ramesh, A.; Nikam, D.; Balachandran, V.N.; Guo, L.; Wang, R.; Hu, L.; Comert, G.; Jia, Y. Cloud-Based Collaborative Road-Damage Monitoring with Deep Learning and Smartphones. *Sustainability* **2022**, *14*, 8682. [CrossRef]
34. Salcedo, E.; Jaber, M.; Requena Carrión, J. A Novel Road Maintenance Prioritisation System Based on Computer Vision and Crowdsourced Reporting. *J. Sens. Actuator Netw.* **2022**, *11*, 15. [CrossRef]
35. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 7–12 December 2015; Curran Associates, Inc.: Red Hook, NY, USA, 2015; Volume 28.
36. He, K.; Gkioxari, G.; Dollar, P.; Girshick, R. Mask R-CNN. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2961–2969.
37. Cai, Z.; Vasconcelos, N. Cascade R-CNN: High Quality Object Detection and Instance Segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *43*, 1483–1498. [CrossRef] [PubMed]
38. Sun, P.; Zhang, R.; Jiang, Y.; Kong, T.; Xu, C.; Zhan, W.; Tomizuka, M.; Li, L.; Yuan, Z.; Wang, C.; et al. Sparse R-CNN: End-to-End Object Detection With Learnable Proposals. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 14454–14463.
39. Redmon, J.; Farhadi, A. YOLOv3: An Incremental Improvement. *arXiv* **2018**, arXiv:1804.02767.
40. Bochkovskiy, A.; Wang, C.-Y.; Liao, H.-Y.M. YOLOv4: Optimal Speed and Accuracy of Object Detection. *arXiv* **2020**, arXiv:2004.10934.
41. Glenn, J. Ultralytics | Revolutionizing the World of Vision AI. Available online: https://ultralytics.com (accessed on 2 May 2023).
42. Li, C.; Li, L.; Jiang, H.; Weng, K.; Geng, Y.; Li, L.; Ke, Z.; Li, Q.; Cheng, M.; Nie, W.; et al. YOLOv6: A Single-Stage Object Detection Framework for Industrial Applications. *arXiv* **2022**, arXiv:2209.02976.
43. Wang, C.-Y.; Bochkovskiy, A.; Liao, H.-Y.M. YOLOv7: Trainable Bag-of-Freebies Sets New State-of-the-Art for Real-Time Object Detectors. *arXiv* **2022**, arXiv:2207.02696.
44. He, K.; Zhang, X.; Ren, S.; Sun, J. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. In Proceedings of the Computer Vision—ECCV 2014, Zurich, Switzerland, 6–12 September 2014; Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T., Eds.; Springer International Publishing: Cham, Switzerland, 2014; pp. 346–361.
45. Liu, S.; Qi, L.; Qin, H.; Shi, J.; Jia, J. Path Aggregation Network for Instance Segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 8759–8768.
46. MMDetection Contributors OpenMMLab Detection Toolbox and Benchmark. *arXiv* **2018**, arXiv:1906.07155.
47. Dataset: Pothole Detection Using Computer Vision in Challenging Conditions. Available online: https://figshare.com/articles/figure/Potholes_dataset_with_YOLO_annotations/21214400/3 (accessed on 2 May 2023).

48. Everingham, M.; Van Gool, L.; Williams, C.K.I.; Winn, J.; Zisserman, A. The Pascal Visual Object Classes (Voc) Challenge. *Int. J. Comput. Vis.* **2009**, *88*, 303–338. [CrossRef]
49. Adrien, P.; Guyon, I.; Letournel, A.C.; Baró, X.; Escalante, H.; Escalera, S.; Thomas, T.; Xu, Z. CodaLab Competitions: An Open Source Platform to Organize Scientific Challenges. Diss. Université Paris-Saclay, FRA. 2022. Available online: https://cnrs.hal.science/hal-03629462/ (accessed on 2 May 2023).
50. Tao, L.; Zhu, C.; Xiang, G.; Li, Y.; Jia, H.; Xie, X. LLCNN: A Convolutional Neural Network for Low-Light Image Enhancement. In Proceedings of the 2017 IEEE Visual Communications and Image Processing (VCIP), St. Petersburg, FL, USA, 10–13 December 2017; pp. 1–4.
51. Shen, L.; Yue, Z.; Feng, F.; Chen, Q.; Liu, S.; Ma, J. MSR-Net: Low-Light Image Enhancement Using Deep Convolutional Network. *arXiv* **2017**, arXiv:1711.02488.
52. Fan, C.-M.; Liu, T.-J.; Liu, K.-H. Half Wavelet Attention on M-Net+ for Low-Light Image Enhancement. In Proceedings of the 2022 IEEE International Conference on Image Processing (ICIP), Bordeaux, France, 16–19 October 2022; pp. 3878–3882.
53. Lv, F.; Li, Y.; Lu, F. Attention Guided Low-Light Image Enhancement with a Large Scale Low-Light Simulation Dataset. *Int. J. Comput. Vis.* **2021**, *129*, 2175–2193. [CrossRef]
54. Huang, H.; Yu, A.; Chai, Z.; He, R.; Tan, T. Selective Wavelet Attention Learning for Single Image Deraining. *Int. J. Comput. Vis.* **2021**, *129*, 1282–1300. [CrossRef]
55. Wang, G.; Sun, C.; Sowmya, A. Context-Enhanced Representation Learning for Single Image Deraining. *Int. J. Comput. Vis.* **2021**, *129*, 1650–1674. [CrossRef]
56. Dong, H.; Pan, J.; Xiang, L.; Hu, Z.; Zhang, X.; Wang, F.; Yang, M.-H. Multi-Scale Boosted Dehazing Network With Dense Feature Fusion. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 2154–2164.
57. Liu, X.; Ma, Y.; Shi, Z.; Chen, J. GridDehazeNet: Attention-Based Multi-Scale Network for Image Dehazing. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October–2 November 2019; pp. 7313–7322.
58. Liu, J.; Xu, D.; Yang, W.; Fan, M.; Huang, H. Benchmarking Low-Light Image Enhancement and Beyond. *Int. J. Comput. Vis.* **2021**, *129*, 1153–1184. [CrossRef]
59. Liu, W.; Ren, G.; Yu, R.; Guo, S.; Zhu, J.; Zhang, L. Image-Adaptive YOLO for Object Detection in Adverse Weather Conditions. *Proc. AAAI Conf. Artif. Intell.* **2022**, *36*, 1792–1800. [CrossRef]
60. Chu, J.; Zhang, C.; Yan, M.; Zhang, H.; Ge, T. TRD-YOLO: A Real-Time, High-Performance Small Traffic Sign Detection Algorithm. *Sensors* **2023**, *23*, 3871. [CrossRef] [PubMed]
61. Yao, J.; Fan, X.; Li, B.; Qin, W. Adverse Weather Target Detection Algorithm Based on Adaptive Color Levels and Improved YOLOv5. *Sensors* **2022**, *22*, 8577. [CrossRef]
62. Fu, H.; Song, G.; Wang, Y. Improved YOLOv4 Marine Target Detection Combined with CBAM. *Symmetry* **2021**, *13*, 623. [CrossRef]
63. Liu, S.; Wang, Y.; Yu, Q.; Liu, H.; Peng, Z. CEAM-YOLOv7: Improved YOLOv7 Based on Channel Expansion and Attention Mechanism for Driver Distraction Behavior Detection. *IEEE Access* **2022**, *10*, 129116–129124. [CrossRef]
64. Tremblay, M.; Halder, S.S.; de Charette, R.; Lalonde, J.-F. Rain Rendering for Evaluating and Improving Robustness to Bad Weather. *Int. J. Comput. Vis.* **2021**, *129*, 341–360. [CrossRef]
65. Xin, H.; Ye, Y.; Na, X.; Hu, H.; Wang, G.; Wu, C.; Hu, S. Sustainable Road Pothole Detection: A Crowdsourcing Based Multi-Sensors Fusion Approach. *Sustainability* **2023**, *15*, 6610. [CrossRef]
66. Chaturvedi, S.S.; Zhang, L.; Yuan, X. Pay "Attention" to Adverse Weather: Weather-Aware Attention-Based Object Detection. In Proceedings of the 26th International Conference on Pattern Recognition (ICPR), Montreal, QC, Canada, 21–25 August 2022; pp. 4573–4579.