APPLICATIONS OF
MODELLING AND SIMULATION
http://arqiipubl.com/ams eISSN 2600-8084 VOL 9, 2025, 1-11

Road Damage Detection for Autonomous Driving Vehicles using YOLOv8 and Salp Swarm Algorithm

Nik Ahmad Farihin Mohd Zulkifli[1], Zuriani Mustaffa[1*] and Mohd Herwan Sulaiman[2]

[1]Faculty of Computing, Universiti Malaysia Pahang Al-Sultan Abdullah, 26000 Pekan, Pahang, Malaysia
[2]Faculty of Electrical and Electronics Engineering Technology, Universiti Malaysia Pahang Al-Sultan Abdullah, 26000 Pekan, Pahang, Malaysia

[*]Corresponding author: zuriani@umpsa.edu.my

Submitted 27 July 2024, Revised 17 October 2024, Accepted 27 October 2024, Available online 01 January 2025.

Abstract: Road accidents are one of the leading causes of death and serious injury in Malaysia, often resulting from human errors and poor road conditions. Autonomous vehicles aim to reduce accidents by mitigating human errors. Therefore, improving the road damage detection model in autonomous vehicles is crucial for enhancing their decision-making capabilities and reducing road accidents. Finding suitable sets of hyperparameters for this task is time-consuming. Consequently, this paper proposes a method to improve the detection accuracy of You Only Look Once version 8 (YOLOv8) using Salp Swarm Algorithm (SSA) for hyperparameter optimization, focusing on eight key parameters. The model is trained using the Czech data in Road Damage Dataset RDD2022 from the Crowdsensing-based Road Damage Detection Challenge (CRDDC'2022), with 80% of the data used for training and 20% for validation. The YOLOv8n model is trained with SSA on the RDD2022 dataset, specifically using data from India and China, to find the optimal parameters. The model is then retrained using the hyperparameters identified by SSA. The YOLOv8 models optimized using SSA are compared with the original YOLOv8 and other YOLO versions (YOLOv5, YOLOv9, and YOLOv10), demonstrating a 3.5% improvement in accuracy after hyperparameter optimization in detecting road damage.

Keywords: Hyperparameter optimization; Object detection; Salp swarm algorithm; YOLOv8.

1. INTRODUCTION

In Malaysia alone, according to the World Health Organization (WHO), over 22.51% of deaths per 100,000 population are due to road injuries [1]. For instance, in 2019, 567,516 traffic accidents were recorded in Malaysia, resulting in 6,167 deaths [2], [3]. Moreover, human error is a significant factor in road accidents, with 61,807 accidents leading to fatal injuries and deaths recorded from 2011 to 2021 due to human error while driving [4]. Thus, the development and improvement in Autonomous Driving Vehicle were believed to alleviate the total number of road accidents due to human error.

Autonomous driving, or self-driving cars, aims to operate vehicles without human intervention using sensors, cameras, radar systems, and artificial intelligence algorithms [5]. One of the tasks required for autonomous vehicles is detecting objects on the roads such as pedestrians, other cars, and road damage. There have been multiple occurrences of autonomous vehicles involved in road accidents due to failure in perception of what happening on the road [6]. As such, this paper will focus on enhancing the performance of object detection tasks, specifically for detecting road damage. The techniques for object detection are divided into two, traditional and deep learning-based method object detectors. Traditional methods such as the Viola-Jones (VJ) detector [7], Histograms of Oriented Gradients (HOG) [8], and Deformable Part-based Model [9] are not suitable for autonomous vehicles due to their tendency to create excessive numbers of the same proposal and it also struggles with complex datasets like road damages because the window scales are designed manually using low-level visual cues [10].

Deep learning-based methods are separated into two categories which are two-stage detectors and one-stage detectors. Techniques in two-stage detectors were Region-Based Convolutional Neural Network (R-CNN) [11], Spatial Pyramid Pooling Networks [12], Fast R-CNN [13], and Faster R-CNN [14]. These two-stage detectors work by separating the object detection into two stages which are feature proposal network and classification of the object. Even though these methods achieve higher accuracy compared to traditional methods, the computational power required, and the inference speed could not catch up to be used in embedded systems and real-time detection for Autonomous Vehicle [10]. Therefore, one-stage detectors like Single Shot Multi-Box Detector (SSD) [15], You Only Look Once (YOLO) [16], EfficientDet [17], and Retina Net [18] were developed to improve detection speed by achieving lower accuracy.

The use of SSD, YOLO, and RetinaNet in road damage detection has been explored in various ways, leading to significant improvements [19]–[24]. Over the years, YOLO has outperformed other one-stage detectors with their balanced speed and accuracy and easy implementation even in road damage detection [25]–[27]. YOLO has undergone several iterations, with incremental improvements, and YOLOv8 developed by Ultralytics has achieved the best results compared to its predecessors in both accuracy and speed due to its anchor-free mechanism, which enhances its ability to detect smaller objects [28], [29]. This paper utilizes the smallest version, YOLOv8n (nano) developed by Ultralytics because it offers the fastest detection speed while maintaining a respectable accuracy of 37.3 Mean Average Precision (mAP) 50-95 Intersection Over Union (IoU) in validation, with an 80.4 ms detection speed in ONNX runtime on a CPU, and only 3.2 million parameters [29].

In the Global Road Damage Detection 2020 challenge [30], the winning team, IMSC, used a solution based on the Ultralytics-YOLO (u-YOLO) model. Their approach introduced two key techniques: Ensemble Model (EM) and Ensemble Prediction (EP), which, when combined, delivered the best results in terms of accuracy [31]. The EM approach involved training multiple versions of the u-YOLO model, each with different hyperparameter configurations. These different versions of the model were then used together, with predictions from each variant averaged before applying Non-Maximum Suppression (NMS) to remove duplicate or overlapping predictions. The hyperparameters that were fine-tuned during training included image size, optimizer, batch size, and number of epochs. Other important hyperparameters, like learning rate and momentum, were not as easily adjusted because they are continuous variables and harder to optimize without using external resources. Therefore, hyperparameter optimization techniques like SSA could have been applied to further improve the training efficiency of the models.

In other works, YOLOv8 has been used to advance road damage detection for autonomous vehicles, such as in AD-YOLOv8, by employing the GhostConv module to reduce model size while maintaining feature extraction efficiency. Furthermore, the integration of attention mechanisms, such as Multi-Scale Context Aggregation (MCA), enhances feature recognition, especially in complex environments, while technologies like CARAFE optimize spatial modeling for road damage detection, leading to significant improvements in accuracy and speed on embedded devices like the Jetson Nano [32]. Other improvements, such as YOLO with Pyramid Attention Network (YOLO-PAN) model, have shown superior performance in road sign detection by incorporating modules like Laplacian of Gaussian-based Spatial Attention (LSKA) for better localization of small targets and improved pyramid pooling for enhanced feature scaling. These advancements underline the evolving nature of YOLOv8-based models in addressing specific challenges in autonomous driving, such as road damage detection and the need for real-time processing on embedded systems, showcasing their growing importance in real-world deployments [33]. All these works have explored various methods to improve the accuracy of road damage detection, suggesting that optimizing the training process through hyperparameter optimization could enhance performance even further.

There have been rigorous amounts of studies that have been done to improve the performance of object detection models such as changing the backbones, replacing the convolution neural network, adding attention mechanism, and tuning the hyperparameters of the training. Each of the methods is applied to improve certain parts that are lacking in the model. In [34], the author includes a simple attention mechanism (simAM) to the backbone to reduce false positives of the model and changes the traditional convolutional neural network with Ghostconv to reduce the parameters and improve the efficiency of the model. Other than that, the practices of tuning the hyperparameter for the training of the model have also been done in a lot of works [35]–[37].

Traditional methods for tuning the hyperparameters of the model, such as grid search and random search [35], or Bayesian optimization. Victoria *et al*. [36] used Bayesian Optimization to tune their object detection model for the CIFAR-10 dataset. These traditional methods require substantial computational power, making it difficult to explore and find the best possible solution in a short amount of time. This challenge has led to the incorporation of metaheuristic algorithms for hyperparameter optimization which are Swarm Intelligence (SI) and evolutionary algorithms. Evolutionary algorithms such as Genetic Algorithm (GA) [38] and Differential Evolution (DE) [39] use crossover and mutation for their exploration and exploitation which in turn reduce information sharing between each individual in the population compared to swarm intelligence. GA and DE also take longer to converge and require higher computational power to execute.

Thus, the use of SI such as Grey Wolf Optimization (GWO) [40], Whale Optimization Algorithm (WOA) [41], and Particle Swarm Optimization (PSO) [42] are used for hyperparameter optimization. Ling Zhi *et al.* [43] optimized the hyperparameters of an improved YOLOv5 model using a novel hybrid of the WOA and GWO to further improve the model's mAP. Johnson *et al*. [44] applied hyperparameter optimization using hybrid SSA to find the optimal parameters for a multimodal object detector utilizing YOLOv5 and RetinaNet.

PSO is one of the earliest SI algorithms and has been widely used in hyperparameter optimization for training Deep Convolutional Neural Networks (DCNN) [45]. The SSA is another metaheuristic approach that has been utilized for optimizing hyperparameters in DCNN [46]. Compared to PSO, SSA is simpler to implement due to its straightforward nature and reduced number of parameters, making the tuning process more manageable. Moreover, SSA's ability to handle noise and outliers effectively enhances its performance in noisy data environments, which is particularly beneficial for complex optimization tasks. Its high efficiency also makes SSA well-suited for large-scale optimization problems, positioning it as a strong alternative to PSO across various applications [47].

The practice of using metaheuristics algorithms like GWO and WOA is very common in optimizing the hyperparameter of object detection model training and it comes with its pros and cons [48]–[50]. In this paper, SSA is chosen as an exploration of a new method for hyperparameter optimization to improve the accuracy of YOLOv8. As such, this paper aims to improve the performance of the YOLOv8 model for road damage detection using hyperparameter optimization with SSA. The model chosen is YOLOv8n (nano), and the parameters selected for optimization include the optimizer, initial learning rate, final learning rate, and momentum. These parameters significantly influence the training process. For instance, the learning rate and momentum determine how quickly the model learns from the data. If the learning rate is too high, the model may learn too fast, which can lead to overfitting. The dataset used in this study is the Czech 2022 data from the Road Damage Dataset 2022 in Crowdsensing-based Road Damage Detection Challenge (CRDDC'2022) dataset from the IEEE Big Data Cup 2022 [51].

The dataset that will be used consists of 2829 train images, 709 test images, 3538 images total images in .jpg, and 1745 files of labels in PASCALVOC XML format.

The primary contribution of this paper is the integration of the YOLOv8 object detection model with the SSA for hyperparameter optimization, aimed at improving road damage detection for autonomous driving vehicles. While YOLOv8 is a cutting-edge model known for its real-time detection capabilities and high accuracy, optimizing its performance through hyperparameter tuning can be a complex and time-consuming task. This paper addresses that challenge by employing SSA, a bio-inspired optimization algorithm, to automate and enhance the hyperparameter tuning process. The resulting framework not only streamlines model training but also improves the accuracy and efficiency of detecting various types of road damage, such as cracks and potholes, which are critical for autonomous driving systems. The paper also provides empirical evidence showing how SSA-YOLOv8 outperforms traditional YOLOv8 and PSO-YOLOv8 in terms of key metrics like precision, recall, and mean average precision (mAP) across multiple categories of road damage. By automating the tuning of important parameters like learning rate, optimizer, and momentum, the paper demonstrates that SSA can significantly enhance YOLOv8's ability to generalize across different road conditions, leading to better performance in real-world autonomous driving applications. This approach can be expanded to other object detection tasks, making it a versatile contribution to the field of machine learning and autonomous systems.

The use of SSA, like any optimization algorithm, comes with significant computational demands, as it requires training multiple object detection models for each member of the population in every iteration. This process, involving repeated model evaluations and adjustments, can significantly increase the overall training time and resource consumption. While SSA demonstrated better results compared to PSO in this study, the actual performance gains may vary depending on the specific application. PSO, despite requiring more external knowledge (such as tuning the C1 and C2 coefficients), can be more efficient in certain scenarios, especially where SSA's broader exploration leads to slower convergence. Therefore, while SSA offers a powerful and flexible optimization framework, its computational costs must be carefully considered, and its effectiveness should be evaluated against other optimization methods like PSO, based on the specific goals and constraints of the task at hand.

The contribution of this paper is summarized as follows:

1. Automatic hyperparameter tuning for YOLOv8 object detection model using the SSA.
2. A novel road damage detection model specifically trained on the Czech RDD2022 dataset, contributing to the field of automated infrastructure maintenance and safety assessment.

The remainder of this paper is structured as follows. Section 2 introduces the system implementation. The implemented methodology is presented in Section 3. Section 4 presents the experimental results and discussion, followed by conclusions in Section 5.

2. SYSTEM IMPLEMENTATION

2.1 Salp Swarm Algorithm

SSA is a swarm intelligence algorithm released in 2017 for solving single-objective problems. SSA simulates the behavior of salps which are marine organisms that move in a coordinated manner to form a chain-like structure [46]. In the algorithm, the salps are divided into two groups: leaders and followers. The leader's salp guides the movement of the entire chain towards the target, while the follower salps update their positions based on the leader's position and their immediate predecessor in the chain.

The single-objective SSA starts by initializing the population of salps randomly within the defined search space, which includes the upper and lower bounds. Then, the fitness of each salp is calculated using the defined objective function. Based on the fitness function, the algorithm evaluates fitness to determine the leader salp, depending on whether the problem is a maximization or minimization problem. For a maximization problem, the leader will be the salp with the highest fitness. The leader salp's position is updated based on a probabilistic approach to exploring new areas in the search space, while follower salps update their positions by partially following the leader. These steps are repeated for a defined number of iterations or until a convergence criterion is met, gradually refining the solutions. The best solution found by the salp swarm is then returned as the optimal or near-optimal solution to the optimization problem.

The simplic'ty of SSA, with its low number of parameters, not only makes it easier to implement but also contributes to its optimization efficiency. With fewer parameters to tune, the algorithm's optimization process becomes more straightforward and less prone to overfitting or complexity issues. Additionally, SSA benefits from its adaptability to different problem types, including single-objective and multi-objective problems, as well as its ability to handle constraints effectively. This adaptability makes it a versatile choice for a wide range of optimization tasks. Furthermore, SSA's scalability allows it to handle large-scale optimization problems efficiently, making it suitable for applications where computational resources are limited. Figure 1 shows the pseudocode of the SSA algorithm.

2.2 Object Detection Model

YOLOv8 [29], released in January 2023 by Ultralytics, was preceded by YOLOv5 [52], also from the same company. YOLOv8 offers different sizes, with smaller variants having fewer parameters and requiring lower computational power, resulting in reduced inference time. YOLO operates as a one-stage detector, integrating feature extraction and classification into a single stage, allowing for faster inference at the expense of some accuracy. Consequently, YOLOv8 is well-suited for real-time applications where speed is crucial.

Initialize the salp population xi (i= 1, 2, ..., n) considering ub and lb
While (end condition is not satisfied)
  Calculate the fitness of each search agent (salp)
  F=the best search agent
  Update c1 by use: $c^1 = 2e^{-\left(\frac{4t}{L}\right)^2}$
  For each salp (x_i)
    if (i ==1)
      Update the position of the leading salp using:

$$x_j^1 = \begin{cases} F_j + c_1\left(\left(ub_j - lb_j\right)c_2 + lb_j\right) c_3 \geq 0 \\ F_j - c_1\left(\left(ub_j - lb_j\right)c_2 + lb_j\right) c_3 < 0 \end{cases}$$

    else
      Update the position of the follower salp using:

$$x_j^1 = \frac{1}{2}\left(x_j^1 + x_j^{i-1}\right)$$

    endif
  endfor
  Amend the salps based on the upper and lower bounds of variables
endwhile
return F

Figure 1. Pseudocode of SSA.

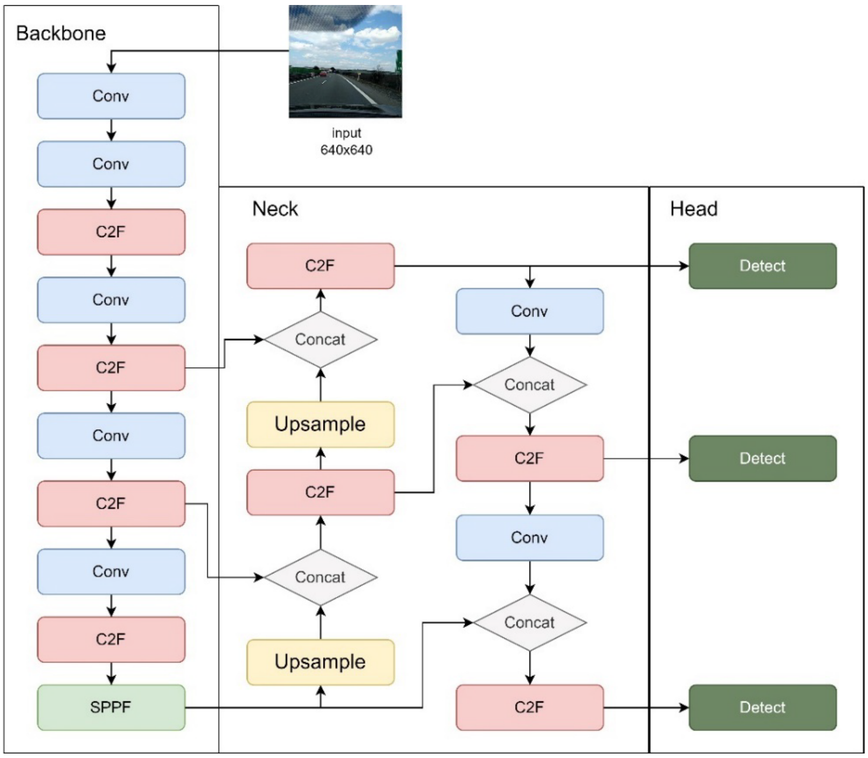

Figure 2. Architecture of YOLOv8.

The anchorless design of YOLOv8 offers several advantages in object detection. By removing predefined anchor boxes, YOLOv8 gains flexibility in handling objects of various scales and aspect ratios, improving detection accuracy. This design simplifies the model architecture, reducing computational complexity and resulting in faster training times and lower resource requirements, making it efficient for real-time applications. Additionally, fewer hyperparameters need tuning, simplifying optimization and enhancing model robustness. Overall, YOLOv8's anchorless design combines accuracy, efficiency, and simplicity, making it an excellent choice for object detection tasks.

Figure 2 shows the architecture of YOLOv8 where the backbone network is tasked with extracting the features of the input images. The backbone network is tasked with extracting the features of the input images. YOLOv8 utilizes the cross-stage partial network (CSPNet) architecture in its backbone network to decrease the computational complexity of the network without compromising on accuracy. The neck serves as a bridge between the backbone network and the detection head. In YOLOv8, the neck incorporates a spatial pyramid pooling (SPP) module. This module employs different pooling sizes to capture features at various scales within the image, enhancing the model's ability to detect objects of different sizes and resolutions effectively. The detection head predicts bounding boxes and class probabilities for each object in the input image. Unlike traditional methods that use predefined anchor boxes, YOLOv8 employs an anchorless design. This design directly predicts the center, width, and height of bounding boxes, allowing for more flexible and accurate object detection.

3. METHODOLOGY

3.1 Dataset Preparation

The data used in this paper is Czech data from Road Damage Dataset 2022, RDD2022, which consists of 2829 train images, 709 test images, and 1745 labels in PASCALVOC XML. There are four types of road damage, namely longitudinal cracks (D00), transverse cracks (D10), alligator cracks (D20), and potholes (D40) are captured in the dataset [51]. Figure 3 shows sample images for road damage categories in the data: (a) Longitudinal cracks (D00), (b) Transverse cracks (D10), (c) Alligator cracks (D20), and (d) Potholes (D40).

Table 1 shows the distribution of images and crack types in the data. The Czech dataset was chosen because it contains the smallest amount of data compared to the datasets from other countries. The labels in PASCAL VOC XML format were converted to YOLO .txt format to be used for training with YOLOv8. The data was then divided into an 8:2 ratio for training and validation, resulting in 1396 training images and 396 validation images.

3.2 Optimization of YOLOv8 Hyperparameter

To explore the ability of SSA in hyperparameter optimization problems to improve the accuracy of the YOLOv8 model, it is crucial to understand the importance of hyperparameter optimization. Hyperparameter optimization is essential because it involves fine-tuning the parameters that control the learning process of a model, such as learning rate, batch size, and network architecture. These hyperparameters significantly impact the model's performance, convergence speed, and generalization ability. Properly optimized hyperparameters can lead to better model accuracy, robustness, and efficiency, ultimately enhancing the overall effectiveness of the YOLOv8 model in various object detection tasks.

Table 2 shows the 4 hyperparameters that will be optimized for the training of YOLOv8 with the number of salps set to 20 and iteration to 7. The chosen hyperparameters are initial learning rate, final learning rate, optimizer momentum, and training optimizer. Learning rate decides how the model updates its weights based on the gradient during training. It controls the step size in moving towards the optimal solution. The initial learning rate sets the pace at the start of training. A high value may cause the model to overshoot the optimal solution, while a low value can lead to slow progress. Optimizing it ensures a balanced learning speed early on. Meanwhile, the final learning rate determines how fine the updates are in the later stages. A lower value helps the model make small, precise adjustments, improving convergence and stability as training finishes.

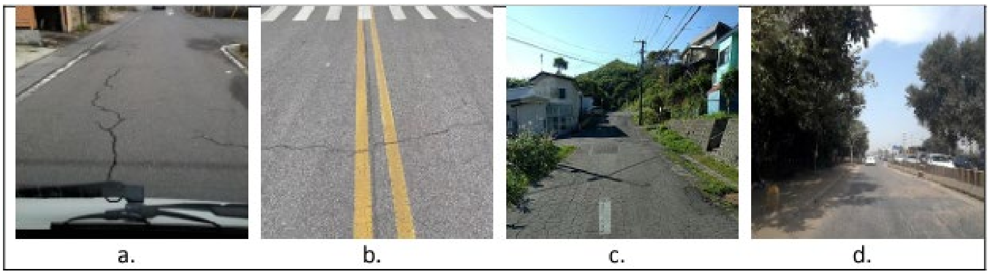

Figure 3. Sample images for road damages categories.

Table 1. Distribution of images and crack types.

Cracks	Name	Instance count
D00	Longitudinal cracks	944
D10	Transverse cracks	399
D20	Alligator cracks	161
D40	Potholes	197

Table 2. Hyperparameters in YOLOv8.

Hyperparameter	Description	Lower bounds	Upper bounds
lr0	Initial learning rate	0.00001	0.01
lrf	Final learning rate	0.01	1
Momentum	Optimizer momentum	0.6	0.98
Optimizer	Training optimizer	SGD or AdamW	SGD or AdamW

Next, momentum helps the optimizer by incorporating information from past gradients, allowing for smoother and faster convergence. It accelerates learning in the right direction while reducing oscillations. A higher momentum helps the model maintain speed through flat regions and avoid getting stuck in local minima, but too much momentum can cause overshooting.

Optimizing momentum ensures the model converges efficiently without instability. Finally, the optimizer determines how the model's weights are updated. SGD (Stochastic Gradient Descent) updates weights based on mini batches, making it straightforward but sensitive to learning rate and momentum tuning. AdamW, on the other hand, adapts learning rates dynamically and incorporates weight decay to mitigate overfitting, making it more robust and less dependent on manual tuning. Optimizing the choice of optimizer, whether SGD for simplicity or AdamW for adaptivity, can significantly affect training performance, especially in complex models.

The implementation of the SSA for hyperparameter optimization in YOLOv8 is illustrated in the flowchart shown in Figure 4. The process begins by randomly initializing 20 salps within predefined upper and lower bounds for each hyperparameter, such as learning rate, batch size, momentum, and weight decay. Each salp represents a potential set of hyperparameters, and these are used to train the YOLOv8n (nano) model using the Czech road damage dataset. Training is conducted for 40 epochs with a batch size of 48 While training, the salps differ in the values of the hyperparameters they represent. This results in different model behaviors some may train faster, some slower, and some may reach higher accuracy, depending on the quality of the hyperparameter values. Once training is completed, the model is evaluated based on its mean average precision at a 0.50 IoU threshold (mAP@50), which serves as the objective function for the SSA. The goal is to maximize mAP@50, allowing the algorithm to rank the salps based on their performance.

The salp with the best score becomes the "leader," and the others are considered "followers." The leader salp's position is updated to explore new regions of the search space, testing new hyperparameter configurations. The followers update their positions relative to the leader and other nearby salps. This process mimics the real-life behavior of salps in a swarm, where the leader drives exploration and the followers make adjustments based on the leader's movement. This process is repeated over 7 iterations, allowing the SSA to efficiently explore a wide range of hyperparameter combinations and converge on an optimal solution. Once the SSA has completed the 7 iterations, it selects the best-performing hyperparameter configuration based on the highest mAP@50 score. This optimal set of hyperparameters is then used to train the YOLOv8n model from scratch, this time for a longer period (100 epochs). This full training process helps ensure the model benefits from the optimal hyperparameter values and performs well across a variety of conditions in the road damage dataset. The final model is then evaluated using not only mAP@50 but also other metrics such as precision, recall, and mAP@50-95.

The use of SSA for hyperparameter tuning in YOLOv8 offers significant advantages. Unlike manual tuning, which can be time-consuming and requires deep domain expertise, SSA automates the process and efficiently navigates the hyperparameter space. Its iterative optimization process ensures that the model is fine-tuned for performance without relying on guesswork or trial-and-error approaches. By balancing the exploration of new hyperparameter configurations with the exploitation of the best-found values, SSA allows YOLOv8 to achieve higher accuracy, better generalization, and more robust performance in road damage detection. This is particularly important for real-world applications such as autonomous driving, where the model must be capable of processing complex environments in real-time while maintaining high accuracy.

3.3 Performance Evaluation Metrics

The models were evaluated using mean average precision (mAP) of 0.5 IoU threshold and 0.5 to 0.95 IoU threshold, as shown in Equation (1). mAP measures the accuracy of the model in detecting objects by calculating the average precision across different recall levels. The other evaluation metrics were precision, P and recall, R as shown in Equations (2) and (3) respectively. Precision indicates the proportion of true positive detections among all detections made given as:

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

where TP is total number of true positive (correctly detected objects), and FP is total number of false positive (incorrectly detected objects). Meanwhile, recall measures the proportion of true positives identified out of all actual positives.

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

where FN is the total number of false negatives (incorrectly detected objects). The average precision, AP is formulated by calculating the area under the precision-recall curve given by Equation (3). $P(R)$ denotes precision as a function of recall R. The integral sums the precision values over all recall levels from 0 to 1, with dR representing a small change in recall, to provide a single value that summarizes the model's performance across all thresholds.

$$AP = \int_0^1 P(R)\, dR \tag{3}$$

On the hand, the mean Average Precision is given by Equation (4) where k representing the number of classes.

$$mAP = \frac{\sum_i^k AP_i}{k} \tag{4}$$

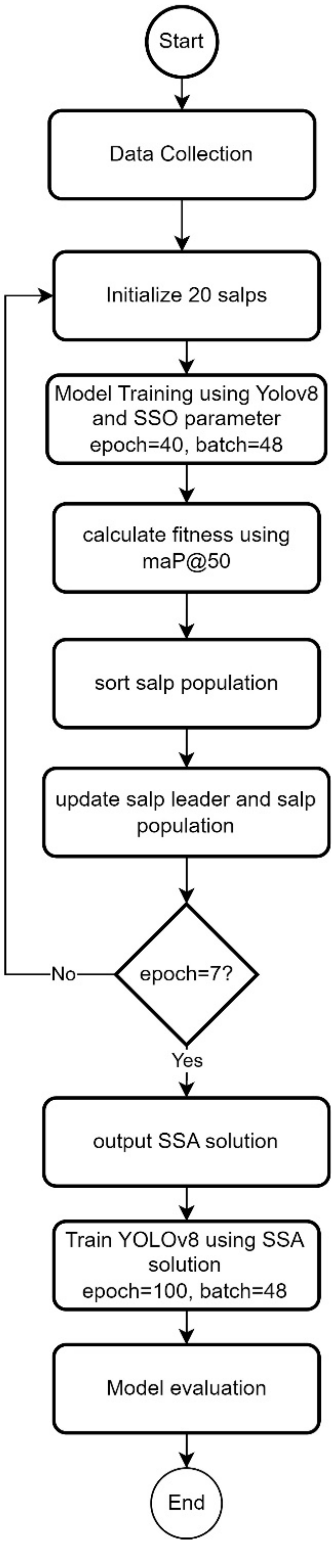

Figure 3. Flowchart of SSA and YOLOv8 implementation.

Table 1. Default and optimized hyperparameters in YOLOv8n.

Hyperparameter	Default	Optimized
lr0	0.00125	0.0038783974451126312
lrf	0.01	0.14310769995725794
Momentum	0.9	0.9260158902286767
Optimizer	AdamW	SGD

Table 2. mAP(0.5)% of SSA YOLOv8n, YOLOv8n, YOLOv5nu, YOLOv9t, YOLOv10.

Hyperparameter	D00	D10	D20	D40	Total
YOLOv5nu	47.0	24.1	36.0	32.0	34.8
YOLOv9-t	47.5	18.3	**40.0**	**32.3**	34.5
YOLOv10n	44.4	18.4	39.5	20.7	30.8
YOLOv8n	**48.2**	11.7	34.8	31.2	31.6
PSO-YOLOv8n	37.9	7.46	35.1	31.0	27.9
SSA-YOLOv8n	45.3	**27.1**	39.6	28.5	**35.1**

4. EXPERIMENTAL RESULTS AND DISCUSSION

The experiment is based on Ubuntu 22.04, using NVIDIA RTX3050 8GB memory GPU. The batch size is 48, the training epochs of 100, and the image size is 640×640. The YOLOv8n model with the SSA hyperparameter is compared with default YOLOv8n, YOLOv5nu [52], YOLOv9-t [53], and YOLOv10n [54].

The default and optimized hyperparameters are shown in Table 3. The optimization of hyperparameters is crucial as it fine-tunes the model to enhance its performance on specific tasks, in this case, road damage detection. The mAP@50 comparison for road damage detection of the SSA YOLOv8n, YOLOv8n, YOLOv5nu, YOLOV9, and YOLOv10 are shown in Table 4. From the results, the performance of SSA-YOLOv8n managed to improve by 3.5% in mAP(0.5)%. This indicates that the optimization process was highly effective for this model, making it the top performer in this comparison.

Secured the second position with a mAP of 34.8%. This is a surprising result, suggesting that YOLOv5nu, despite being an older version, is still competitive in road damage detection tasks. These models followed in performance ranking. The lower accuracy observed for YOLOv9 and YOLOv10 can be attributed to insufficient training iterations and lack of adequate training data. This highlights the importance of extensive training and data sufficiency for achieving high accuracy in object detection models.

The mAP(0.5)% for YOLOv8 dropped by 2.9% and 2.7%, respectively. This decline suggests that the optimized hyperparameters may not be well-suited for detecting these specific damage types, or there could be other factors at play such as over-fitting or suboptimal feature extraction for these categories. Conversely, the mAP(0.5)% for these damage types improved significantly, with a 15.4% increase for D10 and a 4.8% increase for D20. This substantial improvement indicates that the hyperparameter optimization effectively enhanced the model's ability to detect these types of damages, possibly due to better feature representation or more effective learning parameters tailored to these categories.

The analysis reveals that hyperparameter optimization can significantly impact the performance of YOLO models in road damage detection, with some models and damage types benefiting more than others. The improved performance of SSA-YOLOv8n and YOLOv5nu demonstrates the potential of these models when adequately optimized. However, the variability in performance across different damage types (D00, D10, D20, D40) highlights the need for tailored optimization strategies to address specific detection challenges. Future work could focus on further fine-tuning the models and increasing the training data to enhance the performance of YOLOv9 and YOLOv10, ensuring more consistent and reliable detection across all damage categories.

Table 5 shows the result of evaluation metric for default YOLOv8n, PSO-YOLOv8n, and the proposed SSA-YOLOv8n. The table compares the performance of three models YOLOv8n, PSO-YOLOv8n, and SSA-YOLOv8n across four road damage categories (D00, D10, D20, D40) using evaluation metrics like precision, recall, mAP(0.5), and mAP(0.5-0.95). YOLOv8n performs reasonably well in precision for D00 (47.4%) and D40 (34.8%) but struggles with D10 (27.1%). PSO-YOLOv8n shows slight improvements in detecting D20 (46.9%) and D40 (37.6%), but overall performs worse than the default YOLOv8n in terms of general accuracy, with lower mAP scores. SSA-YOLOv8n, however, demonstrates the best performance, significantly improving precision for D10 (41.3%) and achieving the highest overall mAP(0.5) at 35.1%, which reflects its superior accuracy and better generalization to different road damage types.

The training for PSO-YOLOv8n, and SSA-YOLOv8n was conducted with the same number of epochs (7) and population size (20), but the key difference lies in how hyperparameters were optimized. PSO-YOLOv8n required the adjustment of additional parameters, such as C1, C2, and inertia weight, which rely on external knowledge for proper tuning. This external dependency likely contributed to PSO-YOLOv8n's poorer performance compared to the other models, particularly in categories like D00 and D10, and its overall lower mAP scores (27.9% for mAP(0.5) and 9.9% for mAP(0.5-0.95)). The need for expert knowledge to fine-tune PSO's parameters may have hindered its ability to find the optimal model configurations, leading to less effective performance. In contrast, SSA-YOLOv8n, which does not require such external parameters, showed significantly better performance across all metrics. SSA's automatic and efficient optimization process allowed it to achieve the highest precision and mAP scores. This highlights the effectiveness of SSA over PSO in road damage detection, especially for real-world applications where robust performance and ease of implementation are critical. Figure 5 shows the capability of the YOLOv8n model in detecting road damage and classifying its classes.

Table 5. Results of YOLOv8n, PSO-YOLOv8n and SSA YOLOv8.

Model	Indicator	D00	D10	D20	D40	Total
YOLOv8n	Precision (%)	**47.4**	27.1	26.7	34.8	34.0
	Recall (%)	52.4	18.6	34.3	**34.1**	34.9
	mAP(0.5) (%)	**48.2**	11.7	34.8	**31.2**	31.5
	mAP(0.5-0.95) (%)	**18.9**	4.18	12.7	**10.9**	11.7
PSO-YOLOv8n	Precision (%)	45.9	17.4	**46.9**	**37.6**	36.9
	Recall (%)	43.5	14.8	30.8	27.0	29.0
	mAP(0.5) (%)	37.9	7.46	35.1	31.0	27.9
	mAP(0.5-0.95) (%)	13.1	1.86	13.4	8.36	9.18
SSA-YOLOv8n	Precision (%)	45.3	**41.3**	39.5	27.9	**38.5**
	Recall (%)	52.4	**27.9**	**37.1**	32.9	**37.6**
	mAP(0.5) (%)	45.3	**27.1**	**39.6**	28.5	**35.1**
	mAP(0.5-0.95) (%)	15.2	**8.99**	**15.3**	9.8	**12.3**

Figure 5. Road damage detection using SSA YOLOv8n.

5. CONCLUSION

In conclusion, tuning the training hyperparameters for the object detection model has a significant effect on its performance. The use of SSA as the hyperparameter optimizer successfully improved the accuracy of YOLOv8n for road damage detection in autonomous vehicles. By optimizing the critical parameters such as initial learning rate, final learning rate, momentum, and optimizer, the model's performance in mAP(0.5)% improved by 3.5%. Additionally, enhancements were observed in other performance metrics, enabling YOLOv8n to compete effectively with the newer YOLOv9 and YOLOv10 models. This study demonstrates that SSA is highly effective in addressing hyperparameter optimization challenges for object detection models.

SSA-YOLOv8n was specifically trained using a Czech dataset that contained four types of road damages, but it did not include images with blurriness, very small road damages, or variations in lighting and weather. This lack of diversity in the dataset could hinder the model’s ability to generalize well to new or unseen road conditions. Transfer learning techniques or dataset augmentation might be needed to improve generalization, but even then, performance could degrade when exposed to unseen conditions. Addressing this requires careful curation of datasets or continuous model fine-tuning on new road data. To further mitigate these limitations, adjustments to the YOLOv8 architecture may be necessary, such as adding an attention mechanism to help retain detected objects in real-time [22], replacing the front-module detector with a DC module to enhance tiny-object detection [55], or using a Generative Adversarial Network (GAN) to generate more diverse training data, particularly for different lighting conditions [56]. However, these adjustments may also introduce computational overhead, requiring additional resources and optimization to maintain real-time performance for autonomous driving applications.

For future work, further improvements can be achieved by refining the YOLOv8 architecture and utilizing larger, more diverse datasets. These advancements could lead to more accurate and robust road damage detection capabilities for autonomous vehicles, enhancing their safety and reliability. In addition to these structural changes, combining SSA-based hyperparameter optimization with other techniques offers several promising avenues. A hybrid approach with Genetic Algorithms (GA) could balance exploration and exploitation, leading to faster convergence and better results. Incorporating Bayesian optimization could intelligently guide the search toward promising hyperparameter regions, improving efficiency. Ensemble methods that integrate SSA with other metaheuristics like PSO could enhance robustness and reduce overfitting. Extending SSA into multi-objective optimization would allow simultaneous optimization of both accuracy and computational efficiency, which is crucial for real-time road damage detection. Moreover, integrating SSA with Neural Architecture Search (NAS) could optimize not only hyperparameters but also the network architecture itself, further boosting overall model performance [57]. These combined strategies will help address challenges related to computational overhead and model generalization, pushing the boundaries of current object detection technologies.

ACKNOWLEDGEMENTS AND FUNDING

This research was supported by the Universiti Malaysia Pahang Al-Sultan Abdullah (UMPSA), Malaysia under the Distinguished Research Grant RDU223003.

DECLARATION OF CONFLICTING INTERESTS

The authors declare no potential conflicts of interest with respect to the research and publication of this article.

REFERENCES

[1] World Health Organization, *Road Traffic Injuries*, World Health Organization, https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries, 2023 (accessed 01.01.2024).

[2] Kementerian Pengangkutan Malaysia, *Kemalangan dan Kematian Jalan Raya di Malaysia*. https://www.mot.gov.my/my/land/safety/road-accident-and-facilities (accessed 15.05.2024).

[3] Kementerian Pengangkutan Malaysia, *Indeks Kematian Jalan Raya Malaysia*. https://www.mot.gov.my/my/land/safety/malaysia-road-fatalities-index (accessed 15.05.2024).

[4] Polis Diraja Malaysia (PDRM), *Statistik Kemalangan Jalan Raya mengikut jenis kemalangan Dan Kecederaan Mengikut negeri - Set Data - MAMPU*. https://archive.data.gov.my/data/ms_MY/dataset/statistik-kemalangan-jalan-raya-mengikut-jenis-kemalangan-dan-kecederaan-mengikut-negeri (accessed 15.05.2024).

[5] B. R. Kiran et al., Deep reinforcement learning for autonomous driving: a survey, *IEEE Transactions on Intelligent Transportation Systems*, 23(6), 2022, 4909-4926.

[6] P. Penmetsa, P. Sheinidashtegol, A. Musaev, E. K. Adanu, and M. Hudnall, Effects of the autonomous vehicle crashes on public perception of the technology, *IATSS Research*, 45(4), 2021, 485-492.

[7] P. Viola and M. J. Jones, Robust real-time face detection, *International Journal of Computer Vision*, 57(2), 2004, 137-154.

[8] N. Dalal and B. Triggs, Histograms of oriented gradients for human detection, *Proceedings of 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2005)*, San Diego, USA, I, 2005, 886-893.

[9] P. Felzenszwalb, D. McAllester, and D. Ramanan, A discriminatively trained, multiscale, deformable part model, *26th IEEE Conference on Computer Vision and Pattern Recognition (CVPR, 2008)*, Anchorage, USA, 2008, 1-8.

[10] X. Wu, D. Sahoo, and S. C. H. Hoi, Recent advances in deep learning for object detection, *Neurocomputing*, 396, 2020, 39-64,

[11] R. Girshick, J. Donahue, T. Darrell, J. Malik, U. C. Berkeley and J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Columbus, USA, 1, 2014, 5000.

[12] K. He, X. Zhang, S. Ren and J. Sun, Spatial pyramid pooling in deep convolutional networks for visual recognition, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9), 2015, 1904-1916.

[13] R. Girshick, Fast R-CNN, *Proceedings of the IEEE International Conference on Computer Vision*, Santiago, Chile, 2015, 1440-1448.

[14] S. Ren, K. He, R. Girshick and J. Sun, Faster R-CNN: Towards real-time object detection with region proposal networks, http://image-net.org/challenges/LSVRC/2015/results, 2015 (accessed 06.11.2022)

[15] W. Liu et al., SSD: Single shot multibox detector, *Lecture Notes in Computer Science*, 9905, 2016, 21-37.

[16] J. Redmon, S. Divvala, R. Girshick and A. Farhadi, *You Only Look Once: Unified, Real-Time Object Detection*, http://arxiv.org/abs/1506.02640, 2015.

[17] S. Naddaf-Sh, M. M. Naddaf-Sh, A. R. Kashani and H. Zargarzadeh, An Efficient and scalable deep learning approach for road damage detection, *Proceedings of 2020 IEEE International Conference on Big Data (Big Data 2020),* Atlanta, USA*,* 2020, 5602-5608.

[18] T. Y. Lin, P. Goyal, R. Girshick, K. He and P. Dollar, Focal loss for dense object detection, *Proceedings of the IEEE International Conference on Computer Vision*, Venice, Italy, 2017, 2999-3007.

[19] A. Alfarrarjeh, D. Trivedi, S. H. Kim and C. Shahabi, A deep learning approach for road damage detection from smartphone images, *Proceedings of 2018 IEEE International Conference on Big Data (Big Data 2018)*, Seattle, USA, 2018, 5201-5204.

[20] Y. J. Wang, M. Ding, S. Kan, S. Zhang and C. Lu, Deep *Proposal and Detection Networks for Road Damage Detection and Classification*, *Proceedings of 2018 IEEE International Conference on Big Data (Big Data 2018)*, Seattle, USA, 2018, 5224-5227.

[21] J. Zhu, J. Zhong, T. Ma, X. Huang, W. Zhang and Y. Zhou, Pavement distress detection using convolutional neural networks with images captured via UAV, *Automation in Construction*, 133, 2022, 103991.

[22] A. M. Roy and J. Bhaduri, DenseSPH-YOLOv5: An automated damage detection model based on DenseNet and swin-transformer prediction head-enabled YOLOv5 with attention mechanism, *Advanced Engineering Informatics*, 56, 2023, 102007.

[23] V. Mandal, L. Uong and Y. Adu-Gyamfi, Automated road crack detection using deep convolutional neural networks, *Proceedings of 2018 IEEE International Conference on Big Data (Big Data 2018)*, Seattle, USA, 2018, 5212–5215.

[24] L. Ale, N. Zhang and L. Li, Road damage detection using retinanet, *Proceedings of 2018 IEEE International Conference on Big Data, (Big Data 2018)*, Seattle, USA, 2018, 5197-5200.

[25] Y. Zhang, H. Zhang, Q. Huang, Y. Han and M. Zhao, DsP-YOLO: An anchor-free network with DsPAN for small object detection of multiscale defects, *Expert Systems with Applications*, 241, 2024, 122669.

[26] Z. Ge, S. Liu, F. Wang, Z. Li and J. Sun, *YOLOX: Exceeding YOLO Series in 2021*, 1–7, 2021.

[27] A. Bochkovskiy, C. -Y. Wang and H. -Y. M. Liao, *YOLOv4: Optimal Speed and Accuracy of Object Detection*, http://arxiv.org/abs/2004.10934, 2020.

[28] M. Hussain, YOLO-v1 to YOLO-v8, the rise of YOLO and its complementary nature toward digital manufacturing and industrial defect detection, *Machines*, 11(7), 2023, 677.

[29] Ultralytics, *Ultralytics YOLOv8 Docs,* https://docs.ultralytics.com, 2023 (accessed 22.04.2024).

[30] D. Arya et al., Global road damage detection: State-of-the-art solutions, *Proceedings of 2020 IEEE International Conference on Big Data (Big Data 2020)*, Atlanta, USA, 2020, 5533–5539.

[31] V. Hegde, D. Trivedi, A. Alfarrarjeh, A. Deepak, S. Ho Kim and C. Shahabi, Yet another deep learning approach for road damage detection using ensemble learning, *Proceedings of 2020 IEEE International Conference on Big Data (Big Data 2020)*, Atlanta, USA, 2020, 5553-5558.

[32] T. Ho, Target detection algorithm for autonomous vehicles in road scenes, *5th International Conference on Computer Vision, Image and Deep Learning, (CVIDL 2024)*, Zhuhai, China, 2024, 1426-1431.

[33] Y. Ding, Enhanced YOLO optimization study for road sign detection in autonomous driving, *5th International Conference on Computer Vision, Image and Deep Learning (CVIDL 2024)*, Zhuhai, China, 2024, 1316-1320.

[34] Y. Li *et al.*, RDD-YOLO: Road damage detection algorithm based on improved you only look once version 8, *Applied Sciences 2024,* 14(8), 2024, 3360.

[35] J. Bergstra and Y. Bengio, Random search for hyper-parameter optimization, *Journal of Machine Learning Research*, 13, 2012, 281-305.

[36] A. H. Victoria and G. Maragatham, Automatic tuning of hyperparameters using Bayesian optimization, *Evolving Systems*, 12(1), 2021, 217-223.

[37] R. Hossain and D. D. Timmer, Machine learning model optimization with hyper parameter tuning approach, *Global Journal of Computer Science and Technology*, 21(2), 2021, 7-13.

[38] J. H. Holland, Genetic algorithms, *Scientific American*, 267(1), 1992, 66-72.

[39] R. Storn and K. Price, Differential evolution - A simple and efficient heuristic for global optimization over continuous spaces, *Journal of Global Optimization*, 11(4), 1997, 341-359.

[40] S. Mirjalili, S. M. Mirjalili and A. Lewis, Grey wolf optimizer, *Advances in Engineering Software*, 69, 2014, 46-61.

[41] S. Mirjalili and A. Lewis, The whale optimization algorithm, *Advances in Engineering Software*, 95, 2016, 51-67.

[42] G. Yang, A modified particle swarm optimizer algorithm, *8th International Conference on Electronic Measurement and Instruments (ICEMI)*, Xian, China, 2007, 2675-2679.

[43] L. Xu, W. Yan and J. Ji, The research of a novel WOG-YOLO algorithm for autonomous driving object detection, *Scientific Reports*, 13(1), 2023, 1-13.

[44] J. Kolluri and R. Das, Intelligent multimodal pedestrian detection using hybrid metaheuristic optimization with deep learning model, *Image and Vision Computing*, 131, 2023, 104628.

[45] P. Singh, S. Chaudhury and B. K. Panigrahi, Hybrid MPSO-CNN: Multi-level particle swarm optimized hyperparameters of convolutional neural network, *Swarm and Evolutionary Computation*, 63, 2021, 100863.

[46] S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris and S. M. Mirjalili, Salp swarm algorithm: A bio-inspired optimizer for engineering design problems, *Advances in Engineering Software*, 114, 2017, 163-191.

[47] H. Faris, S. Mirjalili, I. Aljarah, M. Mafarja and A. A. Heidari, Salp swarm algorithm: Theory, literature review, and application in extreme learning machines, *Studies in Computational Intelligence*, 811, 2020, 185-199.

[48] H. Xie, L. Zhang, and C. P. Lim, Evolving CNN-LSTM models for time series prediction using enhanced grey wolf optimizer, *IEEE Access*, 8, 2020, 161519-161541.

[49] G. Atteia, N. Abdel Samee, E. S. M. El-Kenawy and A. Ibrahim, CNN-Hyperparameter optimization for diabetic maculopathy diagnosis in optical coherence tomography and fundus retinography, *Mathematics,* 10(18), 2020, 3274.

[50] E. Emary, H. M. Zawbaa and A. E. Hassanien, Binary grey wolf optimization approaches for feature selection, *Neurocomputing*, 172, 2016, 371-381.

[51] D. Arya, H. Maeda, S. K. Ghosh, D. Toshniwal and Y. Sekimoto, *RDD2022: A Multi-National Image Dataset for Automatic Road Damage Detection*, https://datasetninja.com/road-damage-detector, 2022 (accessed 29.05.2024).

[52] R. Kaur and S. Singh, A comprehensive review of object detection with deep learning, *Digital Signal Processing*, 132, 2023,103812.

[53] C. -Y. Wang, I. -H. Yeh and H. -Y. M. Liao, *YOLOv9: Learning What You Want to Learn Using Programmable Gradient Information*, https://arxiv.org/abs/2402.13616v2, 2024 (accessed 25.06.2024).

[54] A. Wang et al., *YOLOv10: Real-Time End-to-End Object Detection*, https://arxiv.org/abs/2405.14458v1, 2024 (accessed 25.06.2024)

[55] H. Lou et al., DC-YOLOv8: Small-size object detection algorithm based on camera sensor, *Electronics,* 12(10), 2023, 2323.

[56] H. Maeda, T. Kashiyama, Y. Sekimoto, T. Seto, and H. Omata, Generative adversarial network for road damage detection, *Computer-Aided Civil and Infrastructure Engineering*, 36(1), 2021, 47-60.

[57] P. Ren, Y. Xiao, X. Chang, P. Huang, Z. Li, X. Chen and X. Wang, A comprehensive survey of neural architecture search: challenges and solution, *ACM Computing Surveys*, 54(4), 2021, 1-34.