King Abdulaziz University
Faculty Of Computing and Information Technology
Department of Computer Science
CPCS432 – Artificial Intelligence II

Team members:

Hanoof Mohammed –1909632
Arwa Mohammed Kulib – 2019574
Ghaida Ahmed Aleryani- 1906952
Arwa Abdulrahman Alamoudi – 2005539
Rahaf Said Alghamdi – 2006609
Najd Khalid Alotaibi - 2006156
Roaa Abdullah Alzahrani - 2005863
Rahaf Omar Jahlan - 19058545

Task Assignment

Name	Task
Roaa Abdullah Alzhrani	- Writing the introduction and dataset sections. - Writing the YOLO Method and Implementation YOLO Model.

Rahaf Saed Alghamdi	- Implementation of the RetinaNet model. - Related work. - Results and result discussion.
Arwa Abdulrahman Alamoudi	- Implementation of the SSD model. - Related work. - SSD experiment and results.
Rahaf Omar Jahlan	- Implementation of the SSD model. - Related work. - Presentation - Writing about SSD Method, its architicture and its pros and cons.
Arwa mohammed kulib	- Writing the efficientDet experiment - Related work. - Implementation of the efficientDet model. - Summary - EfficientDet result result discussion. - conclusion
Ghaida Ahmed Aleryani	- experiment and result of YOLO mode - implementation YOLO Model.
Najd Khalid Alotaibi	- RetinaNet method and experiment. -Implementation of the RetinaNet model. - Related work. - Evaluation metrics and models comparison
Hanouf Mohammed	-related work -the EfficientDet method

Summary

In the exploration of pothole detection models, various approaches employing object detection algorithms such as YOLO, RetinaNet, EfficientDet, and SSD were considered. These models share a common goal of accurately detecting and classifying potholes using deep learning techniques and architectures. However, their differences lie in distinct design choices and architectures, with some models having multiple versions.

For the experimentation phase, a comprehensive dataset from Roboflow was utilized, consisting of 2,742 images categorized into training, validation, and test sets. Transfer learning techniques were applied to four selected models—YOLO, RetinaNet, EfficientDet, and SSD. Each model was described in detail, emphasizing its unique characteristics and applications.

The evaluation metrics, including precision, were used to compare the models fairly. The results indicated that EfficientDet achieved 100% precision, while YOLO, RetinaNet, and SSD exhibited lower precision values. Notably, the precision values were analyzed alongside other metrics, such as recall, to provide a more comprehensive understanding of the models' performance.

In the detailed experiments, the YOLO model was trained with a transfer learning approach, achieving a precision of 56.8%. RetinaNet demonstrated moderate accuracy with a mean Average Precision (mAP) of 0.55. EfficientDet, despite showcasing 100% precision, faced challenges in recognizing false negatives, emphasizing the need for a balanced precision-recall trade-off. SSD, chosen for its real-time capabilities, proved suitable for pothole detection.

The summary concludes with the importance of striking a balance between precision and recall for effective model performance. Further adjustments, such as tuning thresholds and experimenting with architectures, were suggested for model refinement. The analysis of false negatives and real-world testing under diverse conditions was emphasized to enhance the models' overall effectiveness in pothole detection.

1. Introduction

Computer vision is a field in AI that enables computers to understand and process visual data. Computer vision is concerned with extracting information about a scene by analyzing images of that scene. It has numerous applications in areas such as document processing, remote sensing, radiology, microscopy, object detection, industrial inspection, and robots. Object detection is a crucial aspect of computer vision and has undergone significant advancements in recent years. In this context, object detection refers to the task of locating instances of real-world objects within an image. It involves determining both the presence of a specific object and its spatial location within the image.

The main idea of our project is to utilize computer vision techniques, specifically object detection models, to detect road potholes. Road safety is severely threatened by potholes, as they can cause automobile damage, lead to collisions, and potentially result in serious or even fatal injuries to individuals. Traditional methods of identifying potholes, such as manual inspection and visual surveys, require a significant amount of time and effort. Additionally, these methods often yield inaccurate results, particularly in large or remote areas. Today, with recent advancements in deep learning and computer vision, it has become possible to detect road potholes more effectively and efficiently.

Related work

The research paper presented a novel deep-learning approach for the automated identification of different road surface conditions, specifically focusing on the detection of potholes caused by obstacles or driver actions in a collective sensing environment. The researchers focus on the examination and implementation of unique deep learning models, namely convolutional neural networks (CNN) , LSTM algorithms. Their study was conducted with true data, and the outcomes showed a promising exactness in tackling the two problems. The convolutional neural network (CNN) achieves an accuracy of 98%, whereas the Last Short-Term Memory (LSTM) model achieves an accuracy of 78%.

One article makes the notable endeavor of utilizing an AI kit, OAK-D, as an edge platform to deploy deep learning models for real-time pothole detection. The models were evaluated both on an image dataset with varied road conditions and real-time video captured from a moving vehicle. Detailed comparisons of numerous models and object-detection frameworks were conducted, with the intention of identifying the most effective out of them. Tiny-YOLOv4 proved to be superior, achieving a 90% detection accuracy and a frame rate of 31.76 FPS.

This paper addresses the challenge of road maintenance and the increasing number of potholes in countries like India. Manual road maintenance processes are time-consuming, labor-intensive, and prone to human errors. Therefore, there is a need for a cost-effective automated system for pothole

detection. The paper focuses on applying deep learning techniques, specifically Convolutional Neural Networks (CNNs), for pothole detection. However, due to the lack of available pothole datasets in India, the authors have created a new dataset consisting of 1,500 images of Indian roads. This dataset has been annotated and used to train different versions of the YOLO (You Only Look Once) model, including YOLOv3, YOLOv2, and YOLOv3-tiny.
The performance of the trained models is evaluated based on mean Average Precision (mAP), precision, and recall. The Tiny YOLOv3 model with an input size of 608x608 achieved the highest performance in terms of mAP and IoU precision among the models mentioned. It demonstrated accurate object detection and localization, with an mAP@0.25 of 72.12, an mAP@0.50 of 49.71, and an IoU precision of 50.96.

The paper addresses the important issue of pothole detection in road infrastructure maintenance and safety. It proposes a deep learning-based approach using various architectures and compares their performance for real-time pothole detection. The authors gathered a dataset of pothole images captured from a cellphone mounted on a car windshield, which was complemented with other images retrieved from the internet to improve the database size and variety. They employed deep learning algorithms, such as SSD-TensorFlow, YOLOv3-Darknet53, and YOLOv4-CSPDarknet53, for pothole detection. The YOLOv4 achieved high recall, high precision, and mean Average Precision (mAP) of 85.39% for pothole detection. The system has detected potholes up to a hundred meters away and processed images at approximately 20 frames per second (FPS).

This research proposal focuses on addressing the significant issue of potholes and their contribution to road accidents in India. It highlights the limitations of existing methods for pothole detection and presents a new approach that uses a smartphone's camera and location services, combined with the YOLOv7 object detection algorithm, Google API, and accelerometer. The proposed method aims to accurately categorize potholes and count their occurrences. The study emphasizes the potential of YOLOv7 due to its speed and accuracy. The experimentation and results section outlines the model training and evaluation process, showcasing precision-recall curves and the application's front-end. The YOLOv7 algorithm is seen as a highly effective solution for pothole detection, and the future scope includes mapping potholes and recommending routes with fewer road defects based on collected data.

The goal of this study is to apply different YOLO models for pothole detection.(i.e., YOLOv4, YOLOv4-tiny, and YOLOv5s) are experimented to measure their performance involved in real-time responsiveness and detection accuracy using the image set. The image set is identified by running the deep convolutional neural network (CNN) on several deep learning pothole detectors. After collecting a set of 665 images, the set is divided into training, testing, and validation subsets. A mean average precision at 50% Intersection-over-Union threshold (mAP_0.5) is used to measure the performance of models. The study result shows that the mAP_0.5 of YOLOv4, YOLOv4-tiny, and YOLOv5s are 77.7%, 78.7%, and 74.8%, respectively. It confirms that the YOLOv4-tiny is the best fit model for pothole detection.

---

this paper presents LIBSVM is chosen as the SVM tool. LIBSVM has the capability to solve multiclassification problems effectively, a LIBSVM method was employed to distinguish between different types of cracks, including potholes, longitudinal cracks, transverse cracks, and complex cracks. The experimental results demonstrated that the LIBSVM-based detection algorithm outperformed other SVM-based algorithms in terms of performance. One advantage of using LIBSVM is that it eliminates the need for parameter selection, which can be a challenging task when using traditional SVM methods. By utilizing LIBSVM, good classification results can be achieved without the need for manual parameter tuning. In comparison to other methods such as Otsu, edge detection, K-means, and watershed methods, the proposed method in this study exhibited superior segmentation effectiveness and processing efficiency when applied to cement concrete pavement potholes. This can be attributed to the integration of grayscale and texture features in the proposed method, as opposed to relying solely on grayscale or texture information. It is worth noting that the presence of pebbles had minimal impact on the detection of potholes. However, the detection of potholes can be influenced by the presence of sandy soil, as the texture information necessary for accurate detection may be obscured when potholes are covered by sandy soil.

Road surface defects can have a negative impact on safety and comfort, and potholes are a particularly dangerous form of defect. Traditional methods for detecting potholes are manual and visual, which can be time-consuming and inaccurate. In recent years, researchers have developed a variety of automated pothole detection algorithms, but many of these algorithms are based on 2D images and can be susceptible to errors caused by shadows, ambient light, and road surface stains. The authors of this paper propose an automatic pothole detection algorithm that uses 3D data of pavement surfaces. The algorithm is based on computer vision techniques and can identify potholes in road sections, estimating their shape and severity (area, perimeter, and depth). the performance of the algorithm on a dataset of 3D pavement images and found that it achieved a precision of 89.75%, a recall of 92.95%, and an F-score of 91.28%. These results are significantly better than other pothole detection algorithms that have been reported in the literature. In conclusion, the algorithm is a reliable alternative to traditional pothole detection methods, and it can help road administrators to derive data to optimize maintenance and road functionality.

Reference	**Dataset**	**Models**	**Accuracy**
1	Roads images	(CNN), (LSTM)	CNN 98% LSTM 78%

2	Road images and real-time video	Several YOLO versions, chiefly Tiny-YOLOv4	90%
3	1500 Indian roads images	YOLO (You Only Look Once)	Tiny yolov3_ (608*608) mAP@0.25 of 72.12 mAP@0.50 of 49.71 IoU precision of 50.96
4	dataset of pothole images captured from a cellphone mounted on a car windshield, and other pothole images retrieved from the internet	SSD-TensorFlow, YOLOv3-Darknet53, and YOLOv4-CSPDarknet53	YOLOv4 85.39%
5	Approximately 289 images of road surfaces with potholes	YOLOv7 object detection model with E-ELAN	High accuracy demonstrated through precision-recall curves and F1 score analysis.
6	a set of 665 images in 720 × 720 pixels resolution that captures various types of potholes on different road surface conditions	*YOLOv4* *YOLOv4-Tiny* *YOLOv5s*	*YOLOv4-Tiny 78.7%*
7	250 pothole images were collected on four roads	*LIBSVM*	*LIBSVM 97.8%*
8	3D pavement images	*A proposed algorithm that uses computer vision techniques*	*Precision 89.75%* *Recall 92.95%* *F-score 91.28%*

Similarities and differences between the models

Similarities:

- All models are used for pothole detection.
- They employ various object detection algorithms such as YOLO, CNN, LSTM, SSD, and LIBSVM.
- They aim to accurately detect and classify potholes in different settings.
- The models utilize deep learning techniques and architectures for improved performance.

Differences:

- The models have different architectures and design choices.
- Some models, like YOLOv4 and YOLOv3, have multiple versions with different features and performance characteristics.

2. Technical Description

2.1 Dataset

To develop our AI models for this task, we will use an online available dataset provided by Roboflow. Pothole detection dataset is a type of object detection technique consists of 2,742 images, making it a valuable resource for training our computer vision models. It focuses on detecting one class: potholes. The dataset is divided into three sets: the training set, the validation set, and the test set. The training set comprises 92% of the dataset, which corresponds to 2532 images. The validation set, accounting for 5% of the dataset consists of 143 images. Finally, the test set represents 2% of the dataset containing 67 images. To use this dataset, we use a resizing preprocessing technique.

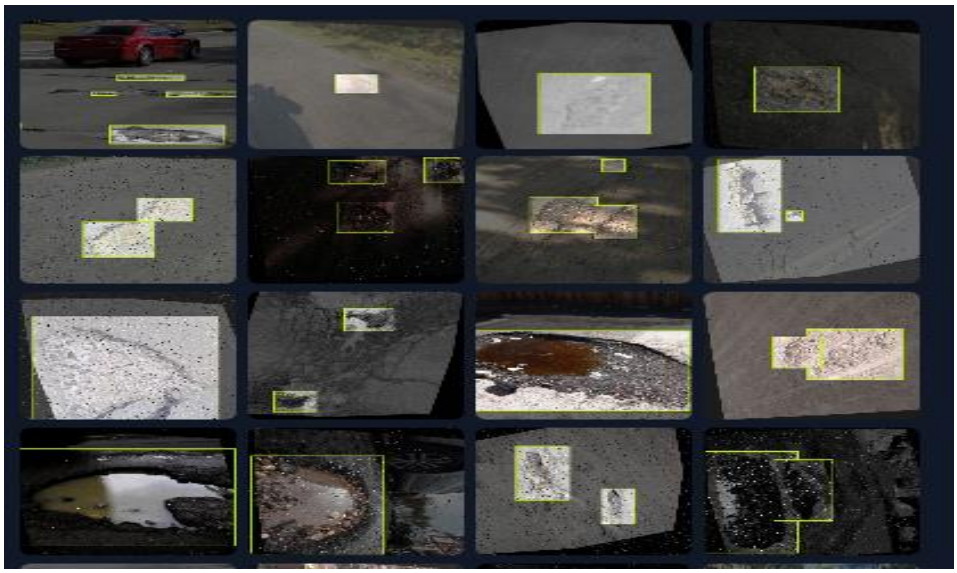


Figure 1: Sample Dataset

2.2. Methods

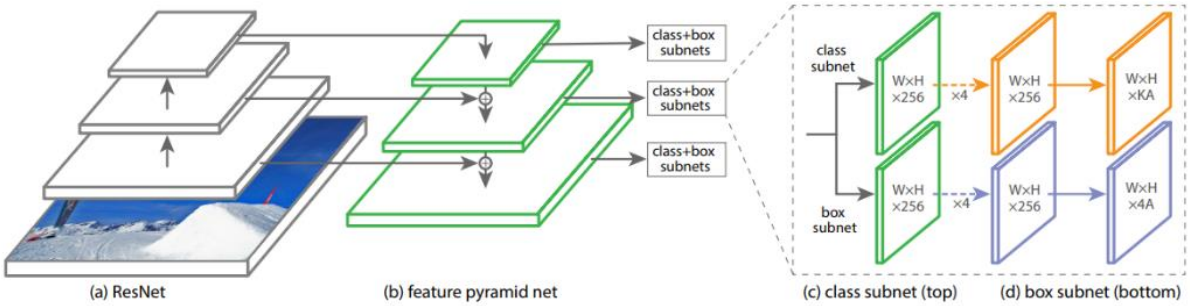
In our experiment on the pothole detection image dataset. We will utilize various pretrained models. Transfer learning techniques will be employed to train these models. The four selected YOLO, RetinaNet, EfficientDet, and SSD model. In this section, models for this experiment are we will describe each of these models.

1. YOLO (You only look once)

YOLO (You Only Look Once) is an object detection algorithm that has gained significant popularity due to its speed and accuracy. It was first introduced by Joseph Redmon et al. Traditional object detection approaches involve using a sliding window or region proposal-based methods to identify objects in an image. These methods can be computationally expensive and time-consuming. YOLO, on the other hand, takes a different approach by formulating object detection as a regression problem. In YOLO, the input image is divided into a grid, and each grid cell predicts a fixed number of bounding boxes along with their corresponding class probabilities and abjectness scores. The bounding boxes are adjusted based on anchor boxes, which are predefined shapes that represent different object sizes and aspect ratios. One of the reasons why YOLO is suitable for pothole detection is its real-time processing capability. Since it performs detection in a single pass through the network, it can achieve near real-time processing rates, making it suitable for applications that require fast object detection. YOLO tends to have lower accuracy in detecting small objects due to its grid-based approach. This can be a challenge when it comes to detecting small-sized potholes. YOLO has undergone several iterations and improvements since its initial release, including YOLOv2, YOLOv3, YOLOv4, YOLOv5, YOLOv6, YOLOv7, and YOLOv8. Each version introduced new architectural enhancements and training techniques to improve accuracy and performance [9].

2. RetinaNet

RetinaNet is a state-of-the-art object detection framework designed to address challenges such as class imbalance and accurate localization of objects in images. It introduces the focal loss to handle class imbalance by assigning different weights to hard and easy examples during training. The model utilizes anchor boxes at various scales and aspect ratios for object localization and incorporates a Feature Pyramid Network (FPN) to capture hierarchical features. With these components, RetinaNet achieves accurate and efficient object detection, making it suitable for a wide range of applications. The architecture of the RetinaNet is illustrated in Fig 2.



2.2.1.1. Why is Using RetinaNet Right?

Using RetinaNet is justified for several reasons. Its focal loss mechanism effectively handles class imbalance, making it robust in scenarios where certain object classes are underrepresented. The model's use of anchor boxes provides flexibility in detecting objects of different scales and shapes, contributing to its versatility. RetinaNet is particularly suitable for scenarios with small or densely packed objects, showcasing its wide applicability. Its state-of-the-art accuracy on benchmark datasets further solidifies its position as a powerful choice for object detection tasks.

2.2.1.2. Pros and Cons of RetinaNet

Pros

- RetinaNet achieves high accuracy in object detection tasks.
- The focal loss mechanism improves model performance on imbalanced datasets.
- RetinaNet is applicable to various domains and scenarios, making it a robust choice for object detection tasks.

Cons

- Training and using RetinaNet can be computationally expensive, especially for large datasets and high-resolution images.
- Achieving optimal performance may require longer training times compared to simpler object detection models.

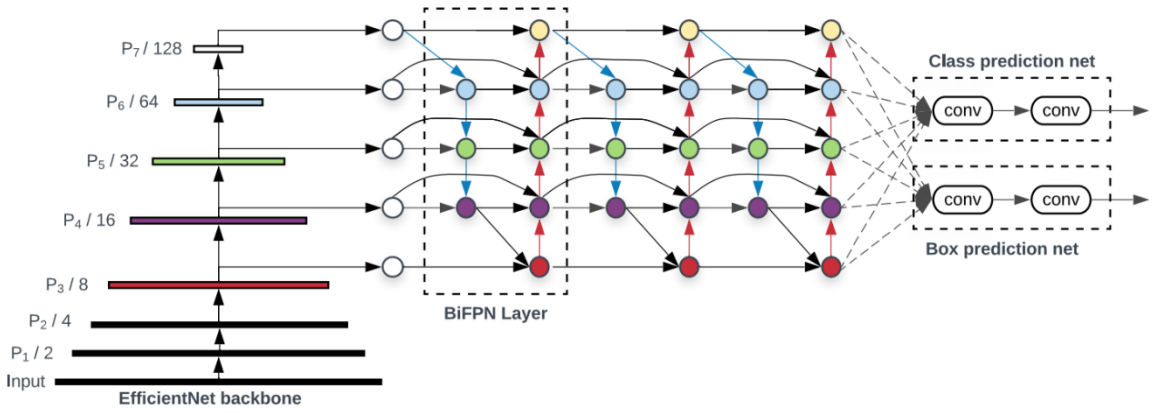
3. EfficienDet:

Developed by a team of researchers at Google Research, EfficientDet gained prominence after being presented in a research paper at the 2020 Conference on Computer Vision and Pattern Recognition (CVPR). This model is renowned for its efficacy in achieving high accuracy in object detection tasks while efficiently utilizing computational resources. EfficientDet falls within the

category of object detection models, a field in computer vision dedicated to identifying and localizing objects in images. Beyond its individual success, EfficientDet models contribute to the broader family of deep learning models widely applied in computer vision, particularly in tasks involving object detection in images and videos. (EfficientDet: Scalable and Efficient Object Detection.,n.d.)

3.1. Why we used EfficienDet?

EfficientDet is selected for its remarkable capacity to strike a harmonious balance between achieving superior accuracy in object detection and maintaining computational efficiency. Built upon the principles of EfficientNet, its innovative architecture enables state-of-the-art performance while efficiently scaling model parameters. This makes EfficientDet especially valuable in scenarios with resource constraints or demanding real-time processing. Widely embraced for its adaptability and effectiveness across diverse deployment contexts, EfficientDet stands out as a preferred solution for various computer vision applications, particularly in tasks involving object detection in both images and videos.in figure below It utilizes EfficientNet as the foundational network, employs BiFPN as the feature network, and shares a class/box prediction network. The repetition of both BiFPN layers and class/box net layers occurs multiple times, adapting to diverse resource constraints (EficientDet :Scalable and Efficient Object Detection,n.d).



3.2. Pros and Cons of EfficientDet:

Cons

Training EfficientDet models demands substantial computational resources, particularly when dealing with larger model variants and extensive datasets. This process often necessitates the use of potent GPUs or TPUs, and the duration can be significant. To attain optimal performance with EfficientDet models, meticulous hyperparameter tuning is

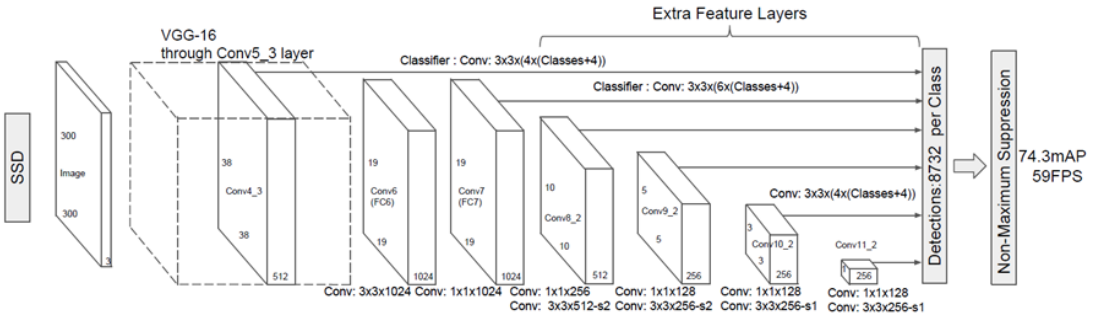
essential. Choices related to learning rates, batch sizes, and other training parameters can significantly influence the model's convergence and ultimate accuracy. Despite EfficientDet models being crafted for efficiency, a delicate balance must be struck between computational efficiency and model accuracy. Depending on the unique demands of your application, finding this equilibrium becomes imperative. (Solawetz, J,2020)

Pros

The use of EfficientDet models has the following advantages.Compared to other object detection models, the EfficientDet models are engineered to attain superior accuracy with a reduced demand on computational resources. It is well known that using EfficientDet models improves parameter efficiency. They can reduce the size of the model and memory footprint by using fewer parameters to achieve competitive performance. Deployment in environments with limited resources benefits from this. A balance is struck between computational efficiency and model accuracy by EfficientDet models. They make use of methods like compound scaling, which scales the depth, width, and resolution of the model all at once. Because of this, the model can operate accurately under a range of resource limitations (EficientDet: Scalable and Efficient Object Detection,n.d).

4. SSD (Single Shot detector)

The Single Shot Multibox Detector (SSD) is a popular object detection model in the field of computer vision. It was designed to achieve real-time object detection in images and videos. Here's an overview of how SSD works, its input and output, and its key components:
The SSD model has two components: the backbone and the SSD head. The backbone, usually based on a pre-trained image classification network like VGG-16, extracts feature maps from the input image. The SSD head, stacked on the backbone, predicts bounding boxes and class scores, detecting objects through multiple convolutional layers.
The SSD divides the image into grids of different sizes and performs object detection for various classes and aspect ratios. It assigns scores to indicate the likelihood of an object's presence. This allows the model to efficiently detect objects of different sizes in the same image. By filtering overlapping bounding boxes, the SSD reduces redundancy and provides a final set of detections.



4.1. Why we used SSD?

SSD is a suitable approach for building a pothole detector due to its real-time capabilities, efficiency in handling various sizes, flexibility with grid sizes, accuracy, and adaptability to different backbone models. Implementing SSD for pothole detection can contribute to road safety and infrastructure maintenance.

4.2. Pros and Cons of SSD:

Pros

- Real-time performance: SSD is efficient and capable of real-time object detection.
- Flexibility with Grid Sizes: SSD can use different grid sizes for different objects, providing flexibility in adapting to various object sizes and aspect ratios
- Non-Maximum Suppression for Accuracy: The non-maximum suppression step in SSD helps filter out redundant and overlapping bounding boxes.

Cons

- Sensitivity to aspect ratios: The performance of SSD can be affected by the predefined aspect ratios of the anchor boxes, which may not perfectly match the aspect ratios of potholes.
- Localization Accuracy for Small Objects: SSD may struggle with precise localization of small objects due to the use of fixed-size default boxes.

3.Experimental Results and Results Discussion

3.1. Evaluation Metrics

To compare the models fairly, the data for each model will be identical and each model will run the same number of epochs, which is 15. Finally, to evaluate the results, we will compare the precision of the models by graphing and illustrating the final results.

Model	Precision
EfficientDet	100%
YOLO	56%
RetinaNet	24%
SSD	22%

3.2. Experiments

3.2.1 Yolo

For the YOLOv5 implementation, we utilized a transfer learning approach with pre-trained weights on a large dataset. The backbone consists of multiple convolutional layers, each followed by a bottleneck CSP (Cross Stage Partial) block to enhance feature extraction. The model was configured with three anchor sets representing different scales, aiding in object detection across varying sizes. The head of the YOLOv5 architecture involves a series of convolutional layers, upsampling, and concatenation operations to merge features from different scales. The final detection layer employs anchor-based detection mechanisms across multiple feature maps. Regarding hyperparameters, the learning rate was set to 0.01, and stochastic gradient descent (SGD) was used as the optimizer with a momentum of 0.9. We adopted a batch size of 16, image size 416*416, and 15 epochs for both the training and testing phases to efficiently process the dataset while minimizing memory consumption. the data (images) pass through the entire YOLOv5 architecture, undergoing multiple convolutional and specialized layers, until the final detection layers, which output the predictions for bounding boxes and class probabilities directly.

3.2.2. RetinaNet Experiment

The experiment aimed to train an object detection model using the RetinaNet architecture, focusing on detecting potholes. Training from scratch was performed using the ResNet-50 architecture, initialized with ImageNet weights. The dataset was taken and then split into train, validate, and test. Each set was preprocessed to have size of 416*416. Once the model was built and the data was ready, we started to train the model. The following criterions were followed in training:

Hyperparameters	Value
Batch Size	2

Epochs	15
Weights	ImageNet Dataset Weights

3.2.3. EfficientDet Experiment

This experiment initiated with the preparation of the road pothole detection dataset, obtained from Roboflow. The dataset was transformed into the TFRecord format, and category labels were defined in the `Potholes_label_map.pbtxt` file. For model training, we selected the `efficientdet-d0` model due to its optimal balance between performance and computational efficiency. Loading pre-trained weights, we crafted a customized pipeline configuration file (`pipeline_file.config`) to fine-tune the model. The training process encompassed configuring and training the model for a specified number of steps using the TensorFlow Object Detection API. The resulting trained model was exported to the `fine_tuned_model` directory for future applications. Following this, a comprehensive evaluation of the model's performance on the test dataset was conducted, involving the computation of key metrics like precision, recall. Lastly, the trained model was deployed to make predictions on a set of test images, visually displaying its adeptness in identifying potential road hazards.

3.2.4. SSD Experiment

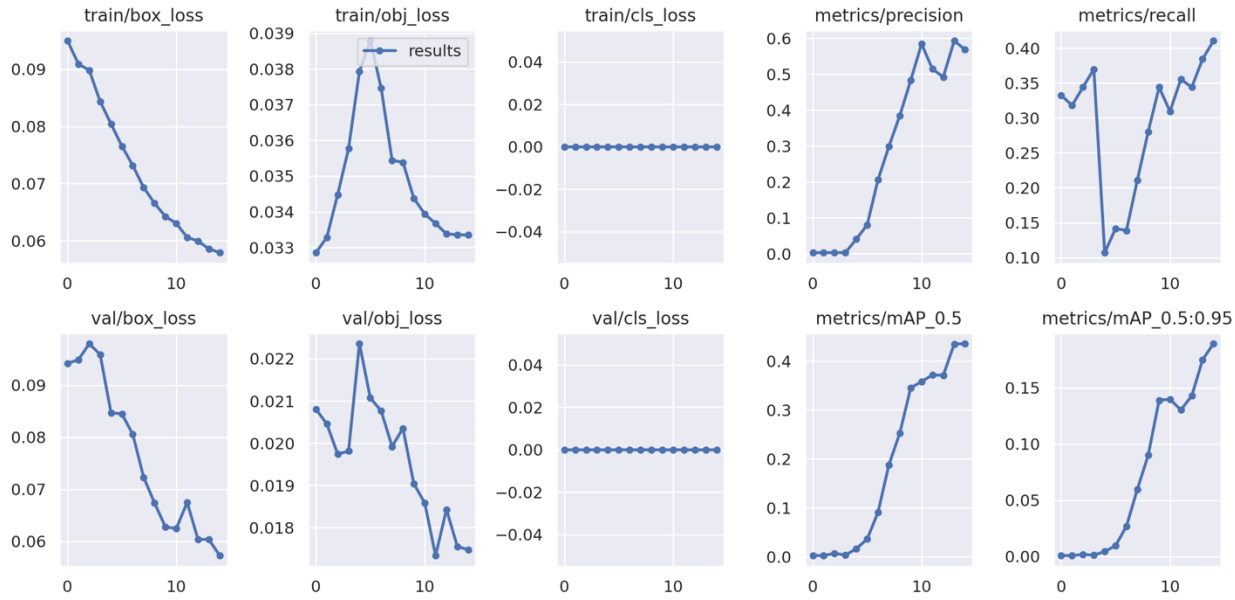
In our experiment, we utilized the pycocotools and torchmetrics Python libraries. We commenced by downloading the dataset in Pascal VOC format and separated the XML annotations from the images into two distinct folders. Subsequently, we executed preprocessing steps, such as thresholding and resizing. The batch size employed was 4, and the model underwent training for 15 epochs. We used ssd300_vgg1 model with its default weights. Following this, we generated bounding boxes and applied data augmentation. Lastly, we evaluated the results of our model using the Mean Average Precision function. Here is a sample of the output:

3.3. Results and Results Discussion

3.3.1 Yolo

After training the model for 15 epoch, we got this result it is nearly the same as the research papers.

Precision (P): 0.568 indicates that when the model predicts an object, it is correct about 56.8% of the time.
Recall (R): 0.41 shows that the model correctly identifies 41% of the relevant objects.
mAP@.5: At an IoU threshold of 0.5, the Mean Average Precision is 0.435, representing the average precision across different object classes. It assesses both the accuracy and the confidence of the model's predictions.
mAP@.5:.95: At varying IoU thresholds from 0.5 to 0.95, the Mean Average Precision is 0.189, providing a broader perspective on the model's performance across various levels of intersection over union.



3.3.2 RetinaNet Results

The model's detection of objects under the sole 'potholes' class achieved a mean Average Precision (mAP) of 0.55, indicating moderate accuracy similar to that of the implementation paper's results. Its precision at 0.24 suggests that a quarter of its pothole detections are correct, while a recall of 0.62 shows it correctly identifies around 62% of actual potholes. These results, while relatively acceptable, point to a higher occurrence of false positives - a byproduct of the lack of non-pothole images.

Validation	
Pothole	45%
Precision	18%
Recall	52%

Testing	
Pothole	55%
Precision	24%
Recall	62%

3.3.3. EfficientDet Results

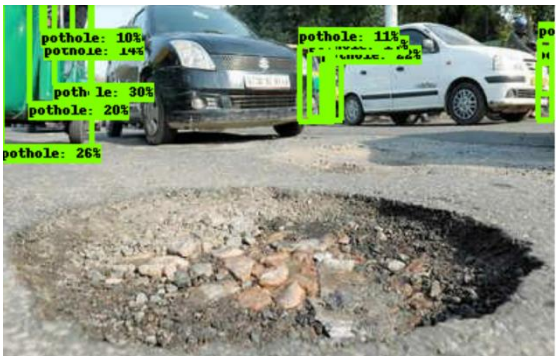
Upon conducting tests, the model exhibited an impressive precision of 100%, signifying a high accuracy in correctly identifying instances of holes. However, a closer examination of the true positives, false positives, and false negatives revealed a more nuanced picture. The model successfully identified 94 instances of holes (true positives), but it failed to recognize 999 actual holes (false negatives).

This skewed precision-recall trade-off suggests that the model places a strong emphasis on achieving precision, aiming for accuracy when it predicts hole detections. Nevertheless, the low recall value (0.086) indicates that the model misses a significant number of true positive instances during detection.

A notable aspect is the model's challenge in recognizing its own mistakes, as evidenced by the false negatives count. When the model fails to identify a hole, it appears to perceive the detection as correct, contributing to the observed precision of 100%.

Moving forward, it is crucial to strike a balance between precision and recall. Adjusting the model's threshold and exploring iterative refinements, such as experimenting with architectures and hyperparameters, can help improve recall without compromising precision significantly. Analyzing false negatives and conducting real-world testing under diverse conditions will contribute to enhancing the model's overall effectiveness in hole detection.

Validation	
Precision	100%
Recall	0.086%



3.3.4. SSD Results

The model's mean Average Precision (mAP) of 0.22 for pothole detection indicates a suboptimal performance, specifically in terms of precision. A precision score of 0.22 implies that less than 25% of the model's predictions for potholes are accurate, reflecting a high rate of false positives. These false positives can undermine the reliability of the model's output.

Validation	
Precision	22%

The low precision score suggests challenges in the model's ability to effectively discriminate between potholes and non-potholes. Possible contributing factors include insufficient or imbalanced training data.

3.4 Comparison Between Results

After getting all the results of the trained models,EfficientDet model gave the best precision in detecting road potholes which is 100%. Compared to the related research, Yolo model achieved higher precision than yolo model of the related research that had precision of 50.96% [3].

4.Conclusion

In conclusion, the application of computer vision techniques, particularly object detection models, for road pothole detection has seen significant advancements, as evidenced by the reviewed literature and our experimental results. Potholes pose a serious threat to road safety, and traditional methods of detection are often time-consuming and prone to inaccuracies. Leveraging deep learning and computer vision technologies has emerged as an effective solution to address these challenges. The reviewed papers and projects have employed various deep learning architectures, including YOLO, RetinaNet, EfficientDet, and SSD, showcasing the diversity of approaches to tackle pothole detection. Each model has its strengths and limitations, and the choice of the model depends on factors such as accuracy requirements, computational efficiency, and the specific characteristics of the road environment. The experimental results of our study with YOLO, RetinaNet, EfficientDet, and SSD models on a pothole detection dataset provided by Roboflow demonstrated the capabilities and trade-offs of each model. Notably, EfficientDet achieved a precision of 100%, highlighting its precision in correctly identifying instances of potholes but it does not detect correctly and does not know that it is a correct detect or not. However, it also showed a challenge in achieving a balanced precision-recall trade-off, emphasizing the need for further refinement and optimization. The similarities among the models lie in their shared goal of accurate pothole detection using deep learning techniques. They employ diverse architectures and design choices, demonstrating the evolution of object detection models over time. On the other hand, differences exist in the specific algorithms, model architectures, and performance characteristics, providing a range of options for addressing the pothole detection problem. Our study contributes to the existing body of knowledge by providing a comparative analysis of multiple object detection models on a standardized pothole detection dataset. The results indicate that there is no one-size-fits-all solution, and the choice of the model should be made based on the specific requirements of the application. Future research directions could involve exploring hybrid approaches that combine the strengths of different models, addressing challenges related to model interpretability and real-time processing, and conducting extensive field testing to evaluate the models in diverse and dynamic road conditions. Additionally, efforts should be made to create more comprehensive and diverse datasets to enhance the generalization capabilities of pothole detection models, especially in regions with unique road characteristics. In conclusion, the field of computer vision for pothole detection

holds great promise, and continued research and development will contribute to improving road safety and infrastructure maintenance.

References

1- Varona, B., Monteserin, A. & Teyseyre, A. A deep learning approach to automatic road surface monitoring and pothole detection. *Pers Ubiquit Comput* **24**, 519–534 (2020). https://doi.org/10.1007/s00779-019-01234-z
2- Asad, M. H., Khaliq, S., Yousaf, M. H., Ullah, M. O., & Ahmad, A. (2022, April 20). Pothole detection using Deep Learning: A real-time and ai-on-the-edge perspective. *Advances in Civil Engineering*. https://www.hindawi.com/journals/ace/2022/9221211/
3- *Deep learning based detection of potholes in Indian roads using YOLO*. (n.d.). IEEE Xplore. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9112424&isnumber=9112371
4- Shaghouri, A. A., Alkhatib, R., & Berjaoui, S. (2021). Real-time pothole detection using deep learning. *arXiv preprint arXiv:2107.06356*. [2107.06356] Real-Time Pothole Detection Using Deep Learning (arxiv.org)
5- *Pothole Detection using CNN and YOLO v7 Algorithm*. (2022, December 1). IEEE Conference Publication | IEEE Xplore. https://doi.org/10.1109/ICECA55336.2022.10009324
6- Park, S., Tran, V., & Lee, D. (2021). Application of various YOLO models for computer Vision-Based Real-Time pothole detection. *Applied Sciences*, *11*(23), 11229. https://doi.org/10.3390/app112311229
7- M. Gao, X. Wang, S. Zhu, and P. Guan, "Detection and Segmentation of Cement Concrete Pavement Pothole Based on Image Processing Technology," *Mathematical Problems in Engineering*, vol. 2020, pp. 1–13, Jan. 2020, doi: https://doi.org/10.1155/2020/1360832.
8- Bosurgi, G., Modica, M., Pellegrino, O., & Sollazzo, G. (2022). An automatic pothole detection algorithm using Pavement 3D data. *International Journal of Pavement Engineering*, 1–15. https://doi.org/10.1080/10298436.2022.2057978
9- J. Redmon, S. Divvala, R. Girshick and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016, pp. 779-788, doi: 10.1109/CVPR.2016.91.

https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7780460&isnumber=7780329

10- T, A. N. (2021, September 25). *Single Shot Detector (SSD) + Architecture of SSD*. OpenGenus IQ: Computing Expertise & Legacy. https://iq.opengenus.org/single-shot-detector/

11- EfficientDet: Scalable and Efficient Object Detection. (n.d.). https://arxiv.org/pdf/1911.09070v7.pdf

12- Solawetz, J. (2020, September 14). A Thorough Breakdown of EfficientDet for Object Detection. Medium. https://towardsdatascience.com/a-thorough-breakdown-of-efficientdet-for-object-detection-dc6a15788b73

13- EficientDet: Scalable and Efficient Object Detection | Object Detection. (n.d.). Www.youtube.com. https://www.youtube.com/watch?v=OsA3zH5NKYc&ab_channel=CodeWithAarohi

Appendix

1. RetinaNet code
2. SSD code
3. Yolo code
4. https://colab.research.google.com/drive/1CLJsRkvYv26DWYK9_ODiPWDXVv8tcScM?usp=sharing

EfficientDet