engineering proceedings

MDPI

Proceeding Paper

Optimizing Pothole Detection in Pavements: A Comparative Analysis of Deep Learning Models †

Tiago Tamagusko and Adelino Ferreira *

Research Centre for Territory, Transports and Environment (CITTA), Department of Civil Engineering, University of Coimbra, 3030-788 Coimbra, Portugal; tamagusko@mail.com

* Correspondence: adelino@dec.uc.pt

† Presented at the Second International Conference on Maintenance and Rehabilitation of Constructed Infrastructure Facilities, Honolulu, HI, USA, 16–19 August 2023.

Abstract: Advancements in computer vision applications have led to improved object detection (OD) in terms of accuracy and processing time, enabling real-time solutions across various fields. In pavement engineering, detecting visual defects such as potholes, cracking, and rutting is of particular interest. This study aims to evaluate YOLO models on a dataset of 665 road pavement images labeled with potholes for OD. Pre-trained deep learning models were customized for pothole detection using transfer learning techniques. The assessed models include You Only Look Once (YOLO) versions 3, 4, and 5. It was found that YOLOv4 achieves the highest mean average precision (mAP), while its shortened version, YOLOv4-tiny, offers the best-reduced inference time, making it ideal for mobile applications. Furthermore, the YOLOv5s model demonstrates potential, attaining good results and standing out for its ease of implementation and scalability.

Keywords: computer vision; object detection; pothole; road pavements; YOLO; deep learning

1. Introduction

This paper investigates state-of-the-art computer vision (CV) techniques in detecting pavement potholes, comparing the performance of various deep learning (DL) models. Object detection (OD) methods, which identify and locate objects in images or videos, have evolved from traditional image processing techniques, such as the Viola-Jones Detector [1] and Histogram of Oriented Gradients, to DL implementations [2,3]. These DL implementations have demonstrated better performance, particularly in complex scenarios, due to their supervised learning approach and the availability data. Hence, the community effort to create massive datasets such as MS COCO [4], PASCAL [5], and IMAGENET [6], has helped the field to evolve. Still, computation power, mainly with GPUs, rapidly increases year by year [7].

One-stage and two-stage detectors are the main categories of DL applications for OD. Two-stage detectors typically exhibit higher accuracy but are slower, while one-stage detectors are faster and more suitable for real-time applications. This article focuses on one of the most famous families of one-stage detectors: You Only Look Once (YOLO) [8]. The YOLO algorithm is a fast and accurate object detection model. It divides input images into grids for simultaneous object detection and classification. Despite lower average precision than some competitors, YOLO's detection makes it ideal for low-latency applications.

This article is organized into four sections, with a brief review of the background, a description of the data and methods used, a presentation of the results, and a conclusion with future recommendations. By comparing the performance of YOLO models, this research aims to determine the most effective method to detect potholes in road pavements, contributing to a safer and well-maintained infrastructure.

check for updates

Citation: Tamagusko, T.; Ferreira, A. Optimizing Pothole Detection in Pavements: A Comparative Analysis of Deep Learning Models. *Eng. Proc.* **2023**, *36*, 11. https://doi.org/10.3390/engproc2023036011

Academic Editor: Hosin (David) Lee

Published: 30 June 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

2. Data and Methods

This study aims to identify the best model for pothole detection using YOLO-based implementations. Six deep learning models were compared, including YOLOv3-tiny [9], YOLOv3 [9], YOLOv4-tiny [10], YOLOv4 [10], YOLOv5s [11], and YOLOv5x [11]. All models were pre-trained on the Common Objects in Context (COCO) dataset, and transfer learning was used, so the developed models use the base of previous models adapted to pothole detection.

A dataset created by Rahman Atikur [12] containing 665 road pavement images with labeled potholes was used, with a 70/20/10 split for training, validation, and testing. An example of labeled images can be seen in Figure 1.

Figure 1. Example image of potholes with labels.

This experiment used a computer specifically assembled to perform high-demand computing tasks with the following specifications:

- CPU: AMD Ryzen 9 5950x
- Memory: 32GB DDR4 3000MHz RAM
- GPU: NVIDIA GeForce RTX 3090
- NVIDIA Driver: 510.68.02
- CUDA: 11.6
- OS: Arch Linux, Kernel 5.17.5-arch1-1

Models run on Python version 3.8.13 and Pytorch 1.10.2. Furthermore, instructions for installation are in the repositories of YOLOv3, YOLOv4, and YOLOv5. In addition, the customized models were trained using Pytorch and Darknet frameworks, specifically YOLOv3 and YOLOv5 for Pytorch and YOLOv4 for Darknet. The base models used in this study are as follows: YOLOv3-tiny (Pytorch, yolov3-tiny.pt), YOLOv3 (Pytorch, yolov3.pt), YOLOv4-tiny (Darknet, yolov4-tiny.conv.29), YOLOv4 (Darknet, yolov4.conv.137), YOLOv5s (Pytorch, yolov5s.pt), and YOLOv5x (Pytorch, yolov5x.pt).

Likewise, the same training hyperparameters were used for all models, namely:

- Batch size: 16
- Epochs: 3000
- Image size: 416
- Patience: 100 (for Pytorch base models)

No data augmentation technique was used, but the default parameters for each model were maintained.

3. Results

The models were compared based on their mean average precision (mAP) and time to infer an image (Figure 2). The goal is to find a highly precise model for detecting and locating potholes while maintaining a short inference time. YOLOv4, YOLOv4-tiny, and YOLOv5s models stood out as the best options.

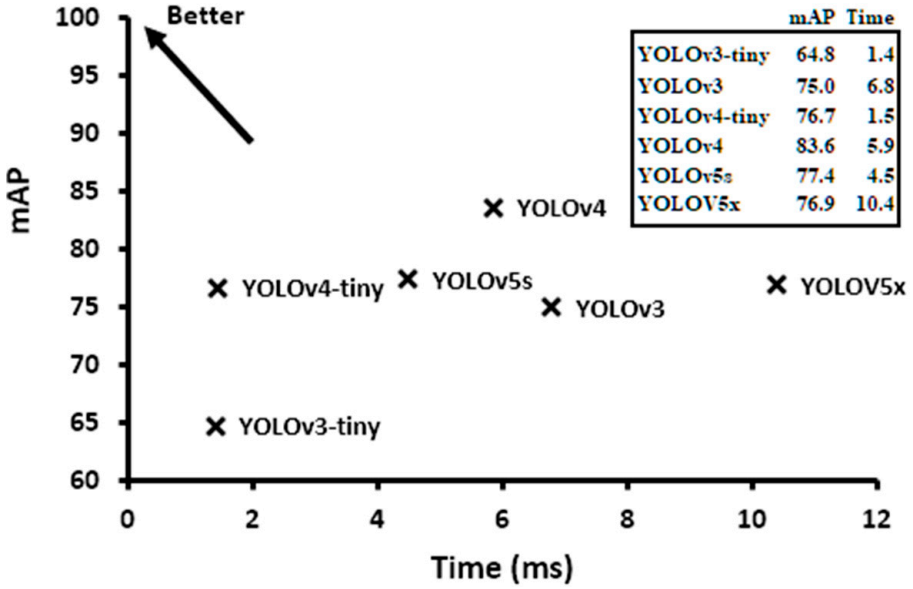

Figure 2. Mean average precision vs. time to infer one image.

YOLOv4 demonstrated greater confidence in predicting small potholes compared to YOLOv5-based models. The detected objects were similar in YOLOv4 versions, and their confidence levels were better than those of YOLOv5 models.

The detailed results are shown in Table 1, with YOLOv4 and YOLOv4-tiny presenting excellent results in terms of mAP, model size, and detection time. In addition, it is believed that mAP could be improved by improving label quality, increasing training data, tuning hyperparameters, and using data augmentation. More precise labeling, such as polygonal segmentation, could also help improve the results.

Table 1. Comparison of YOLO models.

Model	mAP @0.50	Size (MB)	mAP @0.50	Training (s)	Inference (ms)
YOLOv3-tiny	64.8%	16.6	64.8%	779.4	1.4
YOLOv3	75.0%	117.7	75.0%	1002.3	6.8
YOLOv4-tiny	76.7%	22.4	76.7%	249.2	1.5
YOLOv4	83.2%	244.2	83.2%	1254.1	5.9
YOLOv5s	77.4%	13.6	77.4%	512.6	4.5
YOLOv5x	76.9%	165.0	76.9%	1430.7	10.4

Lastly, limitations of this study include the small dataset of 665 images, limited quality of the labels, lack of hyperparameter tuning, and no direct testing of the algorithm's performance in real time. Furthermore, only YOLO implementations were evaluated.

4. Conclusions

The best result obtained for pothole detection in the dataset used was with YOLOv4, reaching a mAP of 83.2%. Still, the implementation with YOLOv4-tiny presents good potential for mobile applications or devices with less computational power. However, training a custom model with YOLOv4 and its usability turns out to be more complex

with the use of the Darknet framework. This becomes an obstacle to putting the model into production and the solution's scalability. On the other hand, version 5 could have better results with some tuning. However, its Pytorch-based implementation is a plus. Consequently, it is recommended to keep the YOLOv4, YOLOv4-tiny, and YOLOv5s models in mind, depending on the application.

As a future research direction, the goal is to expand these custom models to detect more classes, such as alligator cracking, block cracking, longitudinal or transverse cracking, slippage cracks, and rutting. Additionally, more attention will be given to the data, which will be expanded and revised. The ultimate goal is to develop a real-time model capable of detecting various visual defects in road pavements, improving the management of road assets, reducing costs, and improving road safety.

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

Funding: The author Tiago Tamagusko is grateful to the Portuguese Foundation for Science and Technology for the PhD Grant 2020.09565.BD. This research was funded by the Research Center for Territory, Transports and Environment—CITTA (UIDP/04427/2020).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data and models are available at github.com/tamagusko/pothole-detection.

Acknowledgments: The authors would like to thank the support of the Research Centre for Territory, Transports, and Environment CITTA (UIDP/04427/2020) and also ACIV for the presentation of this paper in the conference.

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

References

1. Viola, P.; Jones, M. Robust Real-Time Face Detection. *Int. J. Comput. Vis.* **2001**, *57*, 137–154. [CrossRef]
2. Mittal, U.; Srivastava, S.; Chawla, P. Review of Different Techniques for Object Detection Using Deep Learning. In Proceedings of the Third International Conference on Advanced Informatics for Computing Research, Shimla, India, 15–16 June 2019; Association for Computing Machinery: New York, NY, USA, 2019.
3. Xiao, Y.; Tian, Z.; Yu, J.; Zhang, Y.; Liu, S.; Du, S.; Lan, X. A review of object detection based on deep learning. *Multimed. Tools Appl.* **2020**, *79*, 23729–23791. [CrossRef]
4. Lin, T.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Doll, P. Microsoft COCO: Common Objects in Context. *Eur. Conf. Comput. Vis.* **2014**, *8693*, 740–755.
5. Everingham, M.; Van Gool, L.; Williams, C.K.I.; Winn, J.; Zisserman, A. The pascal visual object classes (VOC) challenge. *Int. J. Comput. Vis.* **2010**, *88*, 303–338. [CrossRef]
6. Fei-Fei, L.; Deng, J.; Li, K. ImageNet: Constructing a large-scale image database. *J. Vis.* **2010**, *9*, 1037. [CrossRef]
7. Pal, S.K.; Pramanik, A.; Maiti, J.; Mitra, P. Deep learning in multi-object detection and tracking: State of the art. *Appl. Intell.* **2021**, *51*, 6400–6429. [CrossRef]
8. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; Volume 1, pp. 779–788. [CrossRef]
9. Redmon, J.; Farhadi, A. YOLOv3: An Incremental Improvement. *arXiv* **2018**, arXiv:1804.02767. [CrossRef]
10. Bochkovskiy, A.; Wang, C.-Y.; Liao, H.-Y.M. YOLOv4: Optimal Speed and Accuracy of Object Detection. *arXiv* **2020**, arXiv:2004.10934. [CrossRef]
11. Jocher, G. "YOLOv5", Ultralytics. Available online: https://github.com/ultralytics/yolov5 (accessed on 23 July 2020).
12. Atikur, R. Annotated Potholes Image Dataset. Available online: https://www.kaggle.com/chitholian/annotated-potholes-dataset (accessed on 15 April 2022).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.