Application of YOLO V8 for front-end loader detection in collision risk zones with reclaimers within a coal yard

Elsirlei O. M. Valim, Fidelis Z. Castro, Gustavo M. Almeida

Control and Automation Engineering Program, Federal Institute of Education, Science and Technology of Espírito Santo (Ifes), Avenida dos Sabiás, 330 - Adress of Laranjeiras, Zip-Code: 29166 - 630, Serra/Espírito Santo, Brazil
cdflei@hotmail.com, rafsat@ifes.edu.br, gmaia@ifes.edu.br

Abstract. In port operations, large machines such as reclaimers are crucial to productivity, transporting up to 8,000 tons per hour. However, due to their size, they present significant risks of collisions with other machines and vehicles, which can cause material and personal damage. This study addresses the development of an intelligent system based on computer vision to detect and alert operators to the presence of equipment in the risk zone. The proposed solution aims to be more accessible and efficient, considering the inflated costs and limitations of current technologies, such as LIDAR and GPS-based systems. Employing the YOLOv8 deep learning algorithm, the system aims to increase operational safety at the Praia Mole Terminal, where older machines lack modern technological upgrades. The research includes the creation of a dataset with images captured by cameras installed on the retrievers, allowing for the detection of objects and the characterization of mobile equipment during port activities.

Keywords: Intelligent System, Collision Avoidance System, Computer Vision, YOLOv8

1 Introduction

Brazilian ports are crucial for cargo transportation, boarding 735 million tons between January and October of [1]. Despite their economic importance, the port environment presents serious safety and health challenges for workers. Between 2016 and 2020, 9,879 accidents were recorded, resulting in thirty-six fatalities and numerous prolonged absences [2]. Accidents involving cargo handling equipment operators are common, ranking fourth in the number of reported Work Accident Communication (CAT) cases [2]. A study at the Guarujá Export Terminal (2018-2020) identified that mobile equipment is associated with severe accidents, including limb loss and fatalities [3]. These data highlight the urgent need to develop and implement safety measures in the port environment, especially for the use of mobile equipment.

At the Praia Mole Terminal (TPM) in the Tubarão Port Complex, Vitória, ES, two reclaimers and eight front loaders are used to move large volumes of cargo. The reclaimers are manually controlled, and their operator cabins have limited visibility, exposing workers to collision and accident risks. These machines can move 2,200 tons per hour [4], and their unavailability can cause significant financial losses. Due to the size and complexity of the reclaimers, collisions can cause catastrophic damage and human fatalities [5]. An incident involving a wheel loader and a reclaimer resulting in material damage, combined with new safety policies, led to the prohibition of simultaneous rehandling in June 2019. This decision caused a 72.19% increase in the time required to clear stockpiles.

To mitigate collision risks and enable the return of simultaneous rehandling activity, different solutions were studied. With recent advances in data processing and artificial intelligence, computer vision has emerged as a more cost-effective alternative for vehicle detection and collision prevention. This research proposes developing a deep learning algorithm based on the YOLOv8 Neural Network to implement an anti-collision system capable of detecting mobile equipment in the risk zone of the reclaimers.

2 Work Methodology

2.1 Field Research

To better understand the environment where the key activity of this research occurs, technical visits were made, and data were collected on the activities. The objective was to identify where the equipment operates, working conditions, current rules and procedures, and basic machine data, such as size, capacities, mode of operation, power source, and restrictions. We interviewed operators to gather their perceptions on the subject. This information was crucial for choosing the algorithm and building the dataset.

It became evident that, at this initial stage, the research should be able to point out systems that identify and alert operators to the presence of wheel loaders nearby, which we defined in this study as the danger zone. At this stage, it was determined that the experimental target object would be wheel loaders or mechanical shovels.

2.2 YOLOv8 – Reasons for Its Selection

The choice had to meet the following requirements: real-time detection of a moving target reliably and quickly, applicable to images from cameras to be installed on the reclaimers. The state-of-the-art research offers extensive literature on object detection algorithms, with considerable prominence for YOLO (You Only Look Once).

Launched in June 2015, YOLO is an object detection model known for its real-time detection capabilities with a single pass through the neural network [6]. Jiang et al. [7] state that the fast-processing speed and strong generalization capability are focal points of YOLO. Safaldin et al. [8] also highlight that YOLO allows for the quick and reliable identification of objects in images. This statement corroborates other conducted research, leading to the decision to use YOLO, with the remaining task of determining which version would be the most appropriate.

YOLOv4 is an improved, faster, and more accurate version that uses the CSPDarknet53 convolutional neural network in its architecture. This version was designed to be an efficient object detector for production systems [9]. YOLOX, developed in PyTorch based on YOLOv3 and incorporating some modifications, achieved cutting-edge results in 2021, balancing speed and accuracy optimally, with 50.1% AP at 68.9 FPS on the Tesla V100 [10].

In January 2023, YOLOv8 was released, which has five variants. It supports multiple vision tasks, such as object detection, segmentation, pose estimation, tracking, and classification. According to Jindal [11] this version of YOLO introduced a new backbone network, Darknet-53, faster and with better accuracy. It is a convolutional neural network with 53 deep layers and can classify images into up to 1,000 categories. Figure 1 presents a comparison of speed and accuracy between YOLOv8 and other versions; the X-axis represents the input parameters and processing latency, and the Y-axis represents the mean average precision (mAP) of the predictions.

Safaldin et al. [8] present a modified version of YOLOv8 that achieved 90% accuracy, a 90% mAP, and maintained a processing speed of 30 frames per second (FPS), with an intersection over union (IoU) score of 80%.

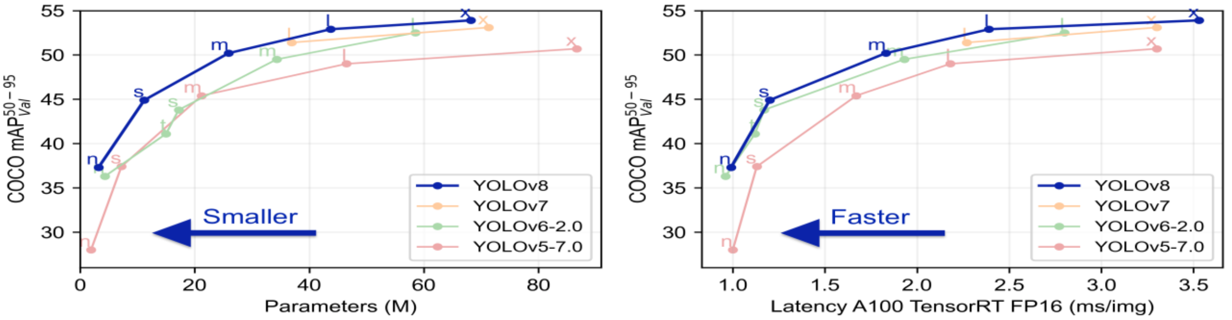

Figure 1. Performance comparison of YOLO versions Source: GitHub.com/Ultralytics (https://github.com/ultralytics/ultralytics)

Rahman et al. [12] proposed using YOLOv8 for detecting static and moving obstacles, highlighting that YOLOv8 outperforms YOLOv7, YOLOv5, SSD, and Faster-RCNN, and the detection accuracy is also higher in YOLOv8. Ko and Rahal compared the performance of YOLOv8 and YOLOv7 and concluded that YOLOv8 is highly superior. Finally, Jindal [11] states that YOLOv8 has a user-friendly API, allowing users to implement the model quickly and easily in their applications. Therefore, based on the studied characteristics and existing applications, we opted to work with YOLOv8n.

2.3 Creating the Dataset Used

The front loaders were chosen as the target object for detection due to the high risk of collision they represent. We conducted various recordings and photographs of these machines, both inside and outside the port area. From these images, we created validation and training files with their respective labels. The image preparation process was supported by the Roboflow website. With each field visit, new images were added; as a result, we obtained five versions.

In the first version, we used 659 images sized 416x416, covering three classes (Machines, Front Loader, and Truck). Only 3% of the images were background, mainly captured during the daytime. In the second version, we added nighttime and background images, reduced it to a single class, and applied data augmentation, totaling 2,036 images, while maintaining the 416x416 size. This first project served as an experimental basis to get to know the site and its functionalities, in addition to being used to test the pre-trained YOLOv8 with the COCO dataset.

In the third version, we increased the number of images to a total of 1,144 images sized 640x640, without using data augmentation and with a single class distributed as follows: 70% for training, 20% for validation, and 10% for testing. This version resulted from improvements made after the initial tests with YOLOv8. The fourth version contains 2,812 images and a single class (Front Loader). In this version, we added more background images, adjusted the bounding boxes, and applied salt and pepper noise to 4% of the images. In the fifth and final version, we have 3,195 images sized 640x640, with salt and pepper noise applied to 4% of the images, gamma exposure adjustment between -10% and +10%, 8.14% of background, and a single class (Front Loader), distributed as follows: 92% for training and 8% for validation.

2.4 Network Configuration and Training

Due to the chosen scenario being within a private company where software installation is restricted, all tests were conducted using the paid version of Google Colaboratory, or simply Colab [13]. It was configured with a Python 3 runtime environment, hardware accelerator, and high RAM. We decided to divide the training into three stages, and to ensure that no data would be lost after each training session, we created a routine to save the data in a Google Drive folder at the end of each stage.

The first stage was to install the created dataset on Google Drive. We used the download feature from Roboflow. The following adjustments were necessary for the dataset to be read correctly: first, we ran the code *dataset_path = '/content/Pá-carregadeira-1'* to identify it. Then, inside the *data.yaml* file, we changed the paths for testing, training, and validation to the respective paths: *test: "/content/Pá-carregadeira-1/test", train: "/content/Pá-carregadeira-1/train", and val: "/content/Pá-carregadeira-1/valid"*.

The second stage was to conduct a simple test using the first version of the dataset from project 1, with the following settings: epochs=10, time=None, patience=100, batch=16, imgsz=640, and the network's native weights. Then, run the algorithm again with the same initial settings but starting from the new weights obtained.

The third stage consists of using the latest version of the dataset from project 2, with the settings: epochs=10, time=None, patience=100, batch=16, imgsz=640, and increasing the number of epochs for 50, 100, 150 until we achieve maximum performance.

3 Results and Discussions

The first test, with 10 epochs using the simplest dataset (first version), demonstrated surprisingly satisfactory results, indicating a trend of improvement. Precision and recall values show fluctuations, but there is a general upward trend, suggesting an improvement in the model's ability to correctly identify objects. The mean Average

Precision (mAP) at different IoU thresholds also shows an upward trend, which is a positive sign of better performance in object detection tasks.

After 50 epochs of training, we observed that both training losses (box_loss, cls_loss, dfl_loss) and validation losses are consistently decreasing, indicating that the model is learning and generalizing well to new data. Precision, recall, and mAP metrics are improving, suggesting increasing performance in object detection and classification, as seen in Figure 2. Although the model is performing well, further training has been conducted, and we chose not to change the hyperparameters.

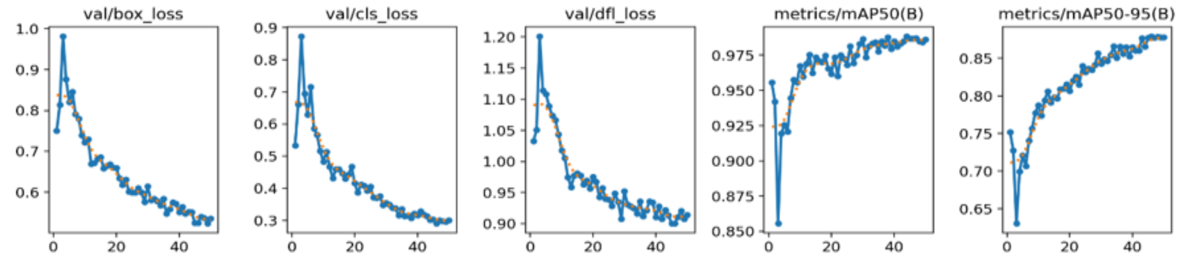

Figure 2. Network results after 50 epochs

After 100 epochs of training, always starting from the weights of the previous training, the network still showed excellent evolution. The precision and recall values are relatively high, especially considering that the ideal value is close to 1. This indicates that the model is correctly identifying most of the positive samples and retrieving a good amount of relevant samples. Oscillations were observed in the validation metrics, especially in recall and mAP50-95. This may suggest that the model is facing specific challenges with some validation samples, possibly due to overfitting or underfitting in certain parts of the data. In Figure 3, we observe the obtained results.

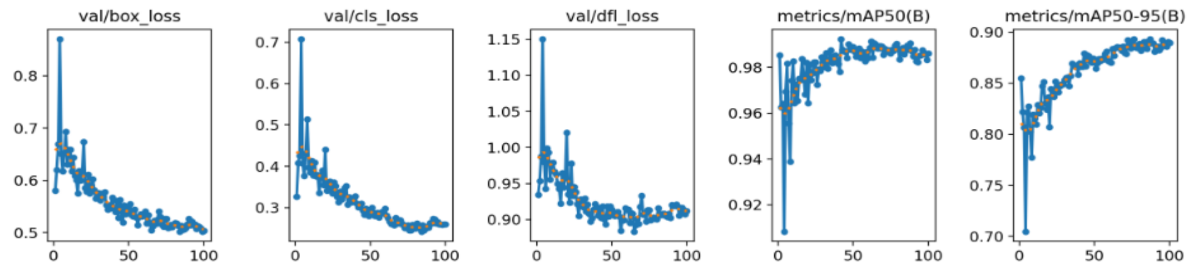

Figure 3. Network results after 100 epochs

After 150 epochs, we observed fluctuations in the training losses over the epochs. The validation losses are generally higher than the training losses, which is expected. Precision and recall vary throughout the epochs but tend to stabilize. The mean Average Precision (mAP) also varies but shows an improving trend over time. The mAP50-95 is particularly important as it evaluates the model's performance across different IoU thresholds. Finally, in Figure 4, the results obtained after 150 epochs are displayed. In summary, the results are still good and indicate that the model is performing well in the object detection task.

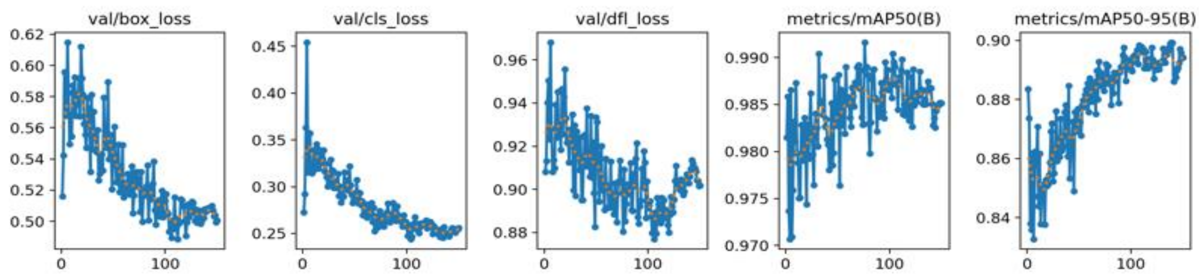

Figure 4. Network results after 150 epochs

When applying the 150-epoch model to real-world scenarios, the results were remarkable, especially in the test with accelerated video of the operation between the reclaimer and the mechanical shovel. Even with a high frame rate, the network managed to maintain excellent accuracy. Figure 5 presents two real-world scenarios: the image on the left shows simultaneous refilling being performed, and the image on the right shows four machines operating on a coal pile.

Figure 5. Test with images from the real-world scenario

These analyzed data indicate that training can be halted at 150 epochs for evaluation and discussions. To potentially achieve even better results, further readings of existing documents and exploration of the most suitable hyperparameters for customization should be considered. Finally, Figure 6 presents a mosaic of images showing the network's performance with each interaction performed on the same image. It is possible to identify that the top center photo has a different annotation; this occurred during the generation of the dataset versions used. Versions 1 and 2 included the annotation "machine," which was later replaced with "mechanical shovel.

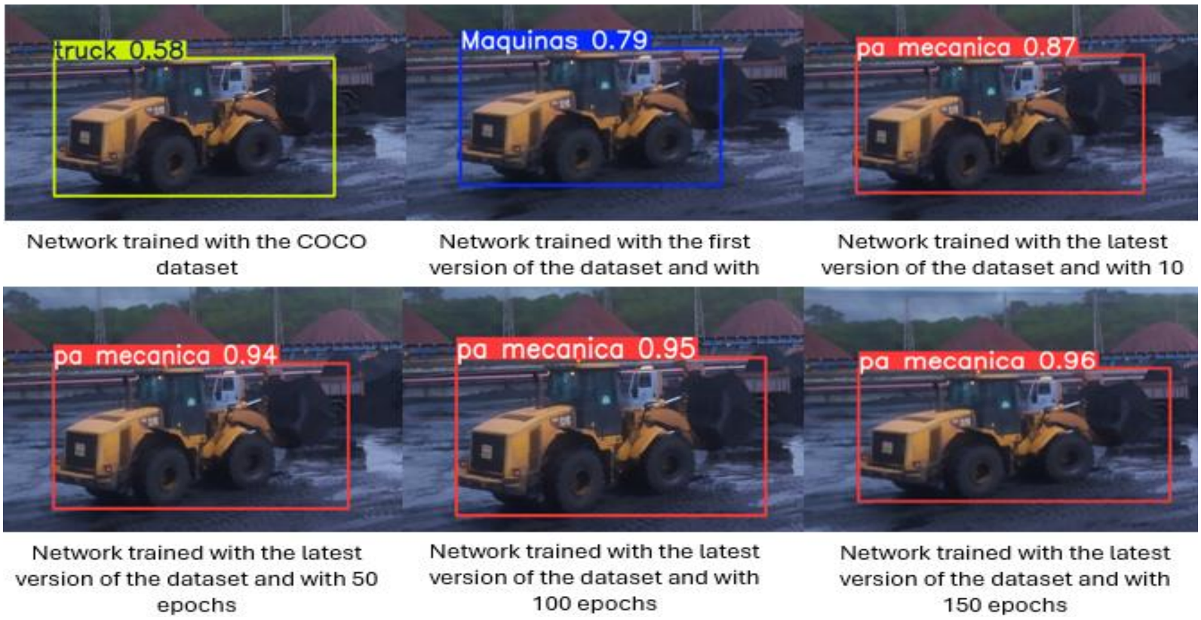

Figure 6. Mosaic illustrating the evolution of detections.

4 Conclusion

In this study, we successfully developed and tested an intelligent system using the YOLOv8 deep learning algorithm to detect front-end loaders in collision risk zones within a coal yard. The system demonstrated significant potential in enhancing operational safety by providing real-time alerts to operators, thereby reducing the risk of accidents. The field research and dataset creation were crucial in tailoring the solution to the specific challenges of the port environment. Through iterative testing and improvement, the model showed promising results in terms of precision and recall, indicating its reliability and effectiveness.

Future work will focus on further refining the algorithm to enhance detection accuracy, especially under varying lighting and weather conditions. Additionally, we aim to integrate this system with other safety measures and explore its applicability to different types of mobile equipment and industrial environments. By continuously improving and adapting this technology, we hope to contribute to safer and more efficient port operations, ultimately benefiting the broader logistics and transportation industry.

Acknowledgements. I would like to express my gratitude and recognition to Dr. Gustavo M. Almeida, who supported me during periods of illness. Without his help, the completion of this work would not have been possible.

Authorship statement. The authors hereby confirm that they are the sole liable persons responsible for the authorship of this work, and that all material that has been herein included as part of the present paper is either the property (and authorship) of the authors, or has the permission of the owners to be included here.

References

[1] Brasil, "Painel Estatístico Aquaviário," ANTAQ - Agencia Nacional de Transportes Aquaviários, 03 janeiro 2023. [Online]. Available: https://web3.antaq.gov.br/ea/sense/index.html#pt. [Accessed 03 janeiro 2023].

[2] R. A. Fuziy, C. A. B. Tarifa, F. L. T. Dinelli, F. R. d. Araújo, M. C. C. Filho, E. C. Lima, J. P. R. R. Teixeira, M. C. d. Silva and S. d. A. Carvalho, "ANÁLISE DE IMPACTO REGULATÓRIO, NORMA REGULAMENTADORA N°29 SEGURANÇA E SAÚDE NO TRABALHO PORTUÁRIO," MINISTÉRIO DA ECONOMIA, Secretaria Especial de Previdência e Trabalho, Brasilia, 2021.

[3] A. R. Machado and J. C. Raymundo, Logística e Gestão da Cadeia de Suprimentos na Área Portuária 1 edição, Belo Horizonte: Poisson, 2021.

[4] MINISTÉRIO DOS TRANSPORTES, PORTOS E AVIAÇÃO CIVIL, UNIVERSIDADE FEDERAL DE SANTA CATARINA, LABORATÓRIO DE TRANSPORTES E LOGÍSTICA , "plano mestre - complexo portuário de - Governo Federal," 10 07 2020. [Online]. Available: https://www.gov.br/infraestrutura/pt-br/centrais-de-conteudo/vp34v1-pdf/@@download/file/vp34v1.pdf. [Accessed 20 05 2024].

[5] M. S. G. D. a. S. K. G. Soumya Bhattacharjya, "Efficient Robust Design Optimization of a Stacker–Reclaimer Structure Under Uncertainty," International Journal of Reliability, Quality and Safety Engineering, p. 4, 15 outubro 2018.

[6] M. A. R. Arthur Kun Ko, "Detecção de Obstáculos com Visão Estéreo para Veículos," 01 janeiro 2023. [Online]. Available: https://repositorio.maua.br/handle/MAUA/501. [Accessed 20 09 2023].

[7] D. E. F. L. Y. C. B. M. Peiyuan Jiang, "A Review of Yolo Algorithm Developments," Procedia computer science, pp. 1066-1073, 03 fevereiro 2022.

[8] N. Z. a. M. M. Mukaram Safaldin, "An Improved YOLOv8 to Detect Moving Objects," IEEE Access, pp. 59782-59806, 25 abril 2024.

[9] T. D. &. G. A. &. J. V. Tembhurne, "Object detection using YOLO: challenges, architectural successors, datasets and applications," Multimed Tools Appl 82, p. 9243–9275, 02 agosto 2022.

[10] D.-M. C.-E. J.-A. R.-G. Juan Terven, "A Comprehensive Review of YOLO Architectures in Computer Vision: From YOLOv1 to YOLOv8 and YOLO-NAS," Deep Learning in Image Analysis and Pattern Recognition, pp. 1680-1716, 20 novembro 2023.

[11] P. Jindal, "YOLOv8 Is Here, and It Gets Better!," 30 janeiro 2023. [Online]. Available: https://pub.towardsai.net/yolov8-is-here-and-it-gets-better-54b12b87e3b9. [Accessed 14 fevereiro 2024].

[12] J. H. R. J. U. a. M. A. S. Shakila Rahman, "Real-Time Obstacle Detection with YOLOv8 in a WSN Using UAV Aerial Photography," Journal of Imaging, 10 outubro 2023.

[13] Google LLC, "Olá, este é o Colaboratory," Google LLC, 26 outubro 2017. [Online]. Available: https://colab.research.google.com/notebooks/welcome.ipynb?hl=pt-BR. [Accessed 30 06 2024].