ISTANBUL TECHNICAL UNIVERSITY

FACULTY OF MANAGEMENT

MANAGEMENT ENGINEERING DEPARTMENT

ISL467E Business Applications with Python

Instructor: Tuncay Özcan

CRN: 12599

Comparative Analysis of Faster R-CNN and YOLOv8 Algorithms For Helmet Detection in Industrial Safety

Term Project Final Report

Mustafa ERCENGİZ - 070210255

Hüseyin SEZEN - 070230334

INDEX

ABSTRACT

Safety of workers is one of the top priorities at a construction site or an industrial area. Recently, due to fast growth in deep learning, object detection models can be employed to identify safety equipment like helmets either in images or video streams. The paper presents a comparison of two deep learning architectures, Faster R-CNN and YOLOv8, applied to the particular task of detecting helmets. In all, each model was trained and further tested on a dataset of 5000 images containing the classes of helmet, head, and person. The models are evaluated based on common metrics such as precision, recall, and mean average precision at 0.5 mAP, and also their inference speed in milliseconds per image (ms/image).

Whereas Faster R-CNN had a high detection accuracy at lower throughputs, the much faster inferences were exhibited with YOLOv8 as it produced higher mAP scores. In general, it should be considered good for near real-time and continuous monitoring. Such a finding reinstates the practical potential of the most modern single-stage detector—YOLOv8—in various safety applications by upscaling several of these challenges that usually include model complexity and speed, apart from dataset balance.

Keywords: Helmet detection, Faster R-CNN, YOLOv8, object detection, deep learning, safety.

1. Introduction

Real-time safety monitoring can be one of life-saving factors in preventing accidents in the workplaces hazards in the industrial and construction environment. Among various safety measures, the use of helmets is a first-line precaution. Object detection algorithms effectively checked compliance with safety rules by automatic methods of inspection. Accurate detection of helmets with speed is indispensable, especially in crowded or dynamic environments where manual surveillance may be highly impractical or prone to errors.

Recent breakthroughs within deep CNNs (LeCun et al., 1998) have marked a new era for powerful object detectors. In this respect, two families of approaches have become dominant:

- Faster R-CNN can be regarded as the best two-stage detector, which gets excellent accuracy by generating proposals first and refining them with a secondary classification head. Though generally appreciated for its solid performance in complicated scenes, heavy computation overhead badly limits Faster R-CNN in real-time scenarios.
- You Only Look Once (YOLO) variants, including YOLO (Redmon et al., 2016) and YOLOv8 (Jocher et al., 2023), belong to a class of one-stage approaches that predict bounding boxes and classify them directly. Based on the idea of speed-oriented design, the YOLO series generally achieves very high throughput. Very recently, YOLOv8 has been proposed by introducing a much stronger backbone and enhanced anchor-free mechanism, which enables its application to industrial scenarios requiring both fast inference and accurate detections.

The paper then performs a comparative study on helmet detection with both Faster R-CNN and YOLOv8, emphasizing their differences. Using a total of 5,000 images containing annotations on classes corresponding to Helmet, Head, and Person, this research determines how these models can identify elements related to classes based on accuracy, recall, mean

average precision at 0.5, and their performance concerning the number of seconds of their operation time. Specific emphasis was made to give prime importance to model efficiency and reliability with regard to helmet detection in an atmosphere of outstanding demands to secure worker safety in real-time conditions.

The rest of the paper is organized as follows: Section 2 provides related work and reviews the literature with respect to recent object detection frameworks. Section 3 describes the data and pre-processing, which covers the composition of the dataset and an exploratory analysis. Section 4 describes the methodology, including model architectures, training parameters, and evaluation metrics. Section 5 summarizes the main findings, pointing out some performance trade-offs in terms of accuracy and speed. Conclusively, Section 6 is about discussion, remark, and future direction that could yield higher performance and make helmet detection advance further in several applications. This will enable stakeholders to understand the strengths and limitations of each detector under the same data and conditions and deploy the most suitable method based on operational requirements, especially when there is a need to balance speed, accuracy, and computational overhead in practical on-site safety surveillance.

2. LITERATURE REVIEW

Object detection has been one of the fundamental milestone in computer vision and has improved with impactful and continuous innovations with the advent of deep learning. The invention of Convolutional Neural Networks by LeCun et al. in 1998 has made it possible to process the data without needing for additional feature extraction, thus leading to the creation of object detection frameworks that are much more advanced. Furthermore the studies prove that in today's digital gathering large sized datasets resulted with higher metrics if the models were created with deep learning algorithms(LeCun, Bengio, & Hinton, 2015). Among those object detection algorithms, Faster R-CNN, proposed by Ren et al. in 2015, and YOLO by Redmon et al. in 2016, became two leading models, as their great efficiency and accuracy could be utilized for many different applications.

Faster R-CNN has employed a two-stage process: first, an Regional Proposal Network(RPN) identifies areas in an image that may contain an object, then a classifier refines these proposals and eventually predicts object classes and bounding box coordinates. In this way, very high accuracy has been achieved, especially for difficult or cluttered scenes, but at great computational cost, often limiting suitability for real-time applications (Ren et al., 2015). In contrast, YOLO is a one-stage bounding box object detection that predicts the bounding box and class probabilities in one step straightforwardly from input images. The design keeps a focus on speed and aims at real-time applications, though it had difficulties with the small or overlapped objects. Successive modifications, including those known as YOLOv3 and YOLOv4, proposed architectural upgrades that allowed increasing both accuracy and speed.

YOLOv8 is the next-generation leap in the series for the YOLO object detection architectures. As it was discussed in the paper "What is YOLOv8: Deep Dive into Next-Generation Object Detector Internal Features," YOLOv8 incorporates state-of-the-art architecture features and new training techniques for better performance and robustness compared to the versions like YOLOv5. Key features developed include a robust backbone for more effective feature extraction, improved anchor-free mechanisms in detecting varied-size objects, and optimizing methods that lessen the computational overhead at almost the same accuracy. For this reason, YOLOv8 is well fitted to real applications that have been requiring speed-accuracy balance, like detecting safety helmets.

These models are very important in workplace safety, detecting whether workers follow the safety regulations regarding helmets. Faster R-CNN is preferred in scenarios where precision is vital, such as crowded industrial sites, while YOLOv8 is ideal for continuous monitoring and dynamic environments because of its real-time capability. The choice of model depends on balancing accuracy and speed, dictated by the specific requirements of the deployment environment. The aim of this work is comparing both models in terms of computational expense and accuracy.

3. DATA

3.1. Dataset

The following study will make use of a dataset containing 5000 images representative of either industrial or construction sites where individuals may or may not be wearing safety helmets. Each image contains three potential object classes annotated:

- Helmet
- Head
- Person

These images mainly focus on whether the safety helmet is present or not. They also contain non-helmeted heads and full persons; thus, multiclass detection can be performed. However, the main aim would go towards the detection of the accurate class of a helmet. The dataset is collected from different open-source repositories and internal images representing a variety in lighting conditions, viewpoints, and backgrounds.

3.2 Exploratory Data Analysis

An initial exploratory analysis on the dataset provides the following useful insights:

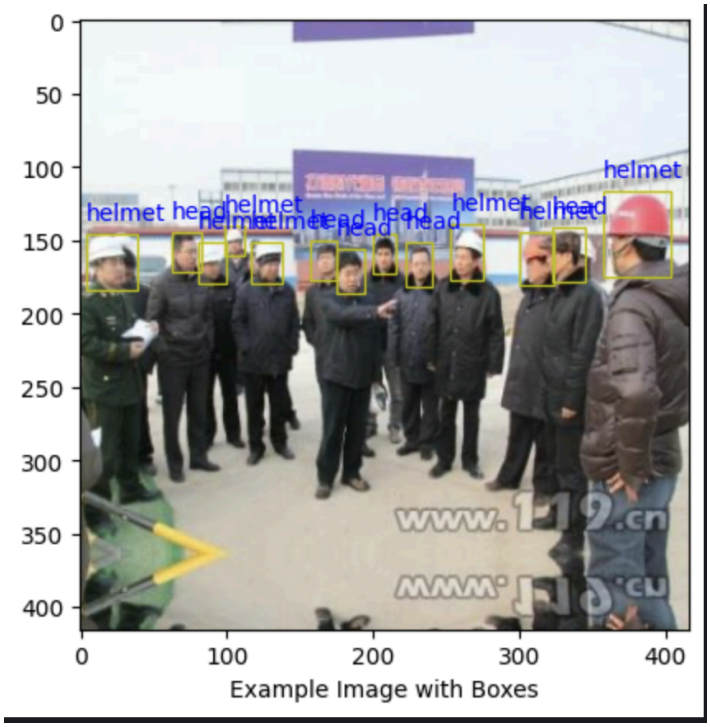
- **Class Distribution:**
    - About 75% of the bounding boxes labeled are helmets.
    - The rest 25% belongs to head and person, with person being the smallest category.

- **Resolution and Quality:**
    - Image files range in size but most are of that medium resolution range of approximately 640x480 or better.
    - Lighting conditions vary from bright sunlight down to dim indoor settings.
- **Missing or Inconsistent Annotations:**
    - None found in the final curated dataset.
    - Then each file was reviewed to make sure the bounding box annotation was complete and had valid coordinates.
- **Preliminary Observations:**
    - The size of the helmet boxes is generally smaller in comparison with the person bounding boxes, for instance, a close-up of the helmet against the full body.
    - Potential class imbalance issues are likely, as helmets far outnumber the other classes.

3.3 Data Visualization

A few data visualization steps were performed to gain insights:

1. **Bounding Box Sizes**:
    - Histograms indicated that helmet bounding boxes occupy a smaller fraction of the image compared to a person.
2. **Sample Image Annotations**:
    - Manually inspected a subset of images to confirm annotation quality.
    - Found that helmet bounding boxes closely track the top portion of a worker's head, typically with fewer occlusions.



4. METHODOLOGY

4.1. Research Aim

This work compares the performance of two object detection models, Faster R-CNN by Ren et al. (2015) and YOLOv8by Jocher et al. (2023) in detecting safety helmets from images.

The primary evaluation metric is to correctly detect the helmets in the construction workers, which ignores data imbalances in the other classes. Though other classes like "head" and "person" are also evaluated by the models, these are secondary to the main task of detecting helmets and also calculated for better training and inference speed comparison.

4.2. Data Preparation

Data Sources and Splitting

- Acquired the dataset containing 5000 images with three target classes of helmet, head, and person, with annotations in Pascal VOC format.

- This was then divided among three subsets:
    - Train: 4000 images.
    - Validation: 500 images.
    - Test: 500 images.
- This split of 80-10-10 ensures that enough data is available for robust training, parameter tuning, and unbiased evaluation.

Annotation Conversion

- Original annotations, which were available in Pascal VOC XML format, were converted to YOLO text files to facilitate the training process in YOLO.
- Therefore, the coordinates of the bounding box changed from the PASCAL VOC format into the form (xmin, ymin, xmax, ymax) to YOLO form as follows:

$$\widehat{x} = \frac{x\ \min\ + x\ \max}{2 \cdot wimg} \qquad \hat{y} = \frac{y\ \min\ + y\ \max}{2 \cdot himg}$$

$$\widehat{w} = \frac{x\ \max\ - x\ \min}{w_{img}} \qquad \widehat{h} = \frac{h\ \max\ - h\ \min}{h_{img}}$$

4.3. Model Architectures and Training

4.3.1. Faster R-CNN

Architecture

- A ResNet-50 backbone pre-trained on the COCO dataset provided initial feature representations (Ren et al., 2015).
- The Region Proposal Network (RPN) and subsequent detection head were fine-tuned to detect the three classes plus background.

Training Procedure

- The model was implemented in PyTorch and trained for 16 epochs.
- Optimizer: Stochastic Gradient Descent (SGD) with momentum $\beta=0.9$ and a weight decay $\lambda=5\times10-4$.
- Learning Rate Strategy: Initial $\eta=0.01$ with Step LR scheduling (step size = 5, gamma = 0.5).

Training Process and Results

Epoch	Training Loss	Validation Loss
1	0,059	0,0644
2	0,0613	0,0613
3	0,0592	0,061
4	0,0576	0,0628
5	0,0557	0,055
6	0,0575	0,0538
7	0,0601	0,0653
8	0,0602	0,0629
9	0,0592	0,0627
10	0,0599	0,0519
11	0,0593	0,0636
12	0,0621	0,0628
13	0,0597	0,0633
14	0,0615	0,0557
15	0,0566	0,0536

16	0,0559	0,0596

The “Training Loss” here is a sum of the classification, regression, and objectness losses for Faster R-CNN during each training epoch.

The “Validation Loss” was computed on the validation set after each epoch; the best model was saved based on the minimum validation loss.

4.3.2. YOLOv8

Architecture:

- Downloaded yolov8m.pt, a medium-scale variant in the YOLO family (Jocher et al., 2023).
- **Training Procedure**:

  Used the Ultralytics library’s command-line interface (CLI) for 20 epochs.

  Optimizer: By default, YOLOv8’s auto setting selected AdamW with an initial learning rate ($\eta \approx 0.0014$) and $\beta = 0.9$.

Training Process and Results

Since YOLOv8 does not provide a direct "val loss," it logs precision (P), recall (R), mAP@0.5, and mAP@0.5–0.95after each epoch. Below is the summary for the *all-class* aggregation (helmet + head + person):

Epoch	Precision (P)	Recall (R)	mAP50	mAP50–95
1	0,906	0,496	0,553	0,327
2	0,893	0,504	0,556	0,336
3	0,937	0,531	0,591	0,354
4	0,918	0,524	0,58	0,358
5	0,942	0,545	0,606	0,381
6	0,938	0,567	0,618	0,381
7	0,941	0,558	0,609	0,386
8	0,926	0,579	0,619	0,392
9	0,948	0,576	0,63	0,403
10	0,93	0,588	0,622	0,391
11	0,951	0,581	0,633	0,407
12	0,948	0,584	0,632	0,408
13	0,954	0,59	0,641	0,413
14	0,95	0,593	0,637	0,414
15	0,939	0,598	0,63	0,414

16	0,952	0,591	0,635	0,414
17	0,943	0,593	0,639	0,425
18	0,947	0,61	0,643	0,426
19	0,958	0,603	0,641	0,423
20	0,959	0,602	0,644	0,429

Interpretation:

- *Precision (P)*: Probability that a predicted box truly belongs to the claimed class.
- *Recall (R)*: Probability that all ground-truth objects of a certain class are successfully found by the model.
- *mAP@0.5*: Mean Average Precision at a 0.5 IoU threshold.
- *mAP@0.5–0.95*: Average of mAP across IoU thresholds from 0.5 to 0.95 (in increments of 0.05).

4.4. Evaluation Metrics

To fairly compare both models' performance, three principal metrics were used, emphasizing the detection of helmets in challenging industrial images.

1. Intersection over Union (IoU)
    - For each predicted bounding box and ground-truth box, IoU is:

$$\text{IoU} = \frac{\text{area of union}}{\text{area of overlap}}$$

    - A predicted box is considered a true positive if IoU ≥ 0.5.
2. Precision & Recall

- For a given class, let TP = true positives, FP = false positives, and FN = false negatives. Then:

$$\text{Precision} = \frac{TP}{TP + FP}, \qquad \text{Recall} = \frac{TP}{TP + FN}.$$

- These metrics capture the trade-off between correctly detected helmets vs. missed or incorrectly predicted instances.

3. Mean Average Precision @ 0.5 (mAP@0.5)
   - Precision-Recall (PR) curves are computed for each class by varying the confidence threshold.
   - Average Precision (AP) is the area under the PR curve:

$$\text{AP} = \int_0^1 p(r), dr$$

     Where p(r) is precision as a function of recall.

   - mAP@0.5 is the mean of these AP values across all classes at IoU ≥0.5.
4. Inference Speed
   - Approximate inference time per image (ms/image) was recorded to gauge real-time feasibility. A rate above 24 frames per second (FPS ≈≈ 42 ms/image) is typically desired for live video analysis (Ren et al., 2015).

5. FINDINGS

5.1. Overall Performance

Upon completing training and evaluation on the test dataset:

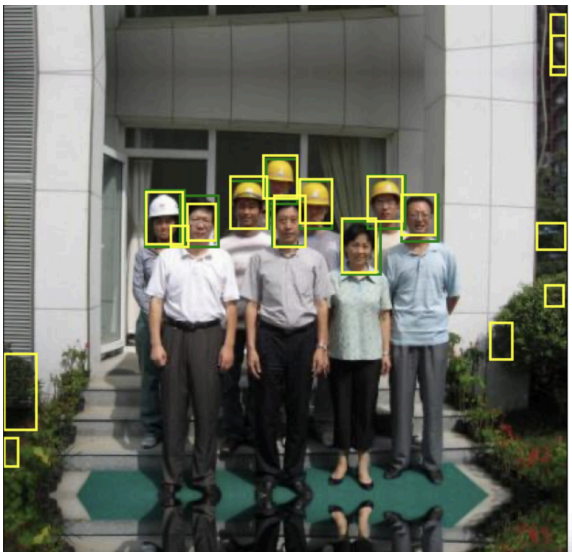
➢ Faster R-CNN:

- mAP@0.5: *~25.34%*
- Precision and Recall:

  Helmet: Precision = 0.7664, Recall = 0.6235

  Head: Precision = 0.5000, Recall = 0.0017

  Person: Precision = 0.0000, Recall = 0.0000
- Inference Speed: *~142 ms per image*



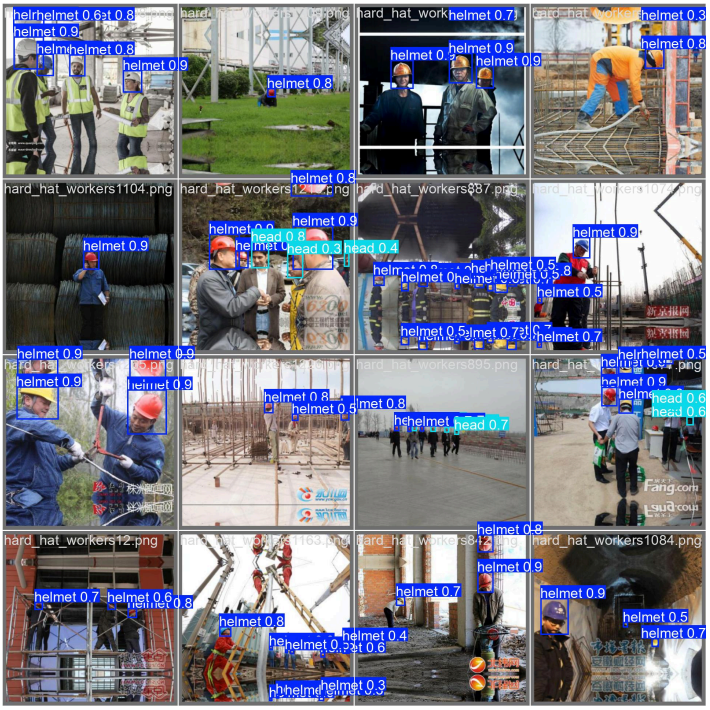
➢ YOLOv8:

- mAP@0.5: *~63.40%*
- Precision and Recall:

  Helmet: Precision = 0.8901, Recall = 0.9383

  Head: Precision = 0.8094, Recall = 0.9213

  Person: Precision = 0.0000, Recall = 0.0000
- Inference Speed: *~10 ms per image*
- *Some Predictions:*



5.2. Observations

1. Accuracy Gains:
    - YOLOv8 demonstrated a significantly higher mAP@0.5 score, indicating better detection capability overall.
    - For tasks where high recall is essential (e.g., ensuring no missed detections of safety helmets), YOLOv8's performance was particularly favorable.
2. Speed vs. Accuracy Trade-off:
    - YOLOv8 ran at ~15 times faster inference speed compared to Faster R-CNN. This speed improvement is critical for real-time or near real-time detection in industrial and surveillance scenarios.
    - Faster R-CNN, while slower, might still be viable if only moderate throughput is required and further hyperparameter tuning or advanced heuristics are used.
3. Class Imbalances:

- Approximately 75% of all bounding boxes belonged to the helmet class, leaving the head and especially the person relatively underrepresented.
- Both algorithms show moderate to low performance for minority classes, suggesting to use a broader dataset in order to perform on other classes than helmet .

4. Potential Improvements:
    - Data Augmentation: Generating synthetic images for underrepresented classes could help improve recall, especially for the `person` class.
    - Hyperparameter Tuning: Further tuning of learning rate, anchor size for YOLO, and multi-scale training may improve the results more.
    - Ensembling: Using the predictions obtained from YOLOv8 in conjunction with those from Faster R-CNN will help in extracting complementary strengths of the models to improve overall results.

---

6 - Conclusion

In the final analysis, considering the chosen dataset for the class 'helmet, the YOLOv8 model showed both accuracy and computational efficiency. Moreover, as the YoloV8 with an Inference Speed of ~10 ms per image proves, that model would be far more applicable in real-life scenarios.

References

1. Bochkovskiy, A., Wang, C. Y., & Liao, H. Y. M. (2020). YOLOv4: Optimal speed and accuracy of object detection. *arXiv preprint*, *arXiv:2004.10934*. https://arxiv.org/abs/2004.10934

2. Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., & Zisserman, A. (2010). The Pascal Visual Object Classes (VOC) Challenge. *International Journal of Computer Vision, 88*(2), 303–338. https://doi.org/10.1007/s11263-009-0275-4
3. Jocher, G., et al. (2023). *ultralytics/ultralytics: v8.0.28 - YOLO and Vision AI in Python*. Zenodo. https://doi.org/10.5281/zenodo.6877093
4. LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature, 521*(7553), 436–444. https://doi.org/10.1038/nature14539
5. LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE, 86*(11), 2278–2324. https://doi.org/10.1109/5.726791
6. Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 779–788. https://doi.org/10.1109/CVPR.2016.91
7. Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster R-CNN: Towards real-time object detection with region proposal networks. *Advances in Neural Information Processing Systems, 28*, 91–99. https://arxiv.org/abs/1506.01497
8. Ren, S., He, K., Girshick, R., & Sun, J. (2017). Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 39*(6), 1137–1149. https://doi.org/10.1109/TPAMI.2016.2577031
9. "What is YOLOv8: A deep dive into internal features of next generation object detector." (2024). *arXiv preprint*, *arXiv:2408.15857*. https://arxiv.org/abs/2408.15857