Object Detection for Safety Attire Using YOLO (You Only Look Once)

Afifuddin Arif Shihabuddin Arip[1], Norazlianie Sazali[1,2,*], Kumaran Kadirgama[3], Ahmad Shahir Jamaludin[1], Faiz Mohd Turan[1], Norhaida Ab. Razak[3]

[1] Faculty of Manufacturing and Mechatronic Engineering Technology, University Malaysia Pahang Al-Sultan Abdullah, 26600 Pekan, Pahang, Malaysia
[2] Centre of Excellence for Advanced Research in Fluid Flow (CARIFF), Universiti Malaysia Pahang Al-Sultan Abdullah, Lebuhraya Tun Razak, 26300, Gambang, Kuantan, Pahang, Malaysia
[3] Faculty of Mechanical and Automotive Engineering Technology, Universiti Malaysia Pahang Al-Sultan Abdullah, 26600 Pekan, Pahang, Malaysia

ARTICLE INFO	ABSTRACT
	Personal protective equipment (PPE) usage is mandated for all employees to prevent workplace accidents and foster a safe and healthy work environment. Using YOLOv8 machine learning and Google Colab's web-based development environment, this research aims to create an immediate detection system for PPE violations in the workplace. By keeping track of PPE compliance, the system is intended to increase workplace safety and prevent accidents. The dataset is collected through a mixture of real-life image gathering and internet datasets. Various images are collected that aim to train the model to detect objects from afar, close, and individually. The research methodology includes a review of the literature, the gathering, pre-processing, and training of models. According to the use of safety helmets, safety shoes, and gloves, there are three different classes of detection based on the bounding box. The system successfully detected the classes with an overall score above 0.8. The safety helmet achieved 0.969, the safety gloves achieved 0.857, followed by the safety vest with 0.887. The findings from this study indicate that the developed system can effectively improve occupational safety and health management. However, there is a detection error factor caused by the lighting and colours. Future research can focus on integrating the system with other work safety systems to provide a comprehensive solution for accident prevention.

1. Introduction

Technology has greatly influenced human history [1-4]. Deep learning and machine learning have recently emerged as transformative fields, significantly advancing our comprehension of computer capabilities [5-7]. The primary goal of this objective is to equip machines with the inherent human capacity for perception, reasoning, and practical application of knowledge [8,9].

Object detection is a crucial aspect of computer vision that has various applications in different industries, particularly in the context of ongoing technological advancements [10-12]. One important

---

* Corresponding author.
E-mail address: azlianie@ump.edu.my

application of personal protective equipment (PPE) is to ensure safety and compliance in various environments [13-15]. This paper explores object detection, specifically the use of the You Only Look Once (YOLO) algorithm for detecting safety attire items like hard hats, safety vests, and goggles in images and videos [16-19]. The ultimate goal is to develop a real-time safety monitoring system that can determine if individuals in hazardous environments comply with the prescribed safety attire protocols. To achieve our objective, we carefully selected and organized a dataset of safety attire images. We use this dataset to train the YOLO algorithm and thoroughly evaluate its performance. Our experimental results confirm that YOLO is effective in detecting safety attire. This highlights its potential as a practical solution for enhancing safety monitoring and compliance in hazardous work environments [20].

The primary responsibility of a company's health and safety officer is to educate employees about safety protocols and assist them in choosing suitable work attire [21]. Challenges arise when individuals disregard safety regulations and work without appropriate attire, especially when they go unnoticed by health and safety officials [22]. The Department of Occupational Safety and Health Malaysia has reported concerning statistics for workplace fatalities in the manufacturing and construction industries during 2018 and 2019. Moreover, research highlights that human errors, suboptimal practices, and equipment failures are significant factors contributing to these unfortunate incidents.

2. Methodology
2.1 Hardware and Equipment

For the creation of the object detection model, this project will harness both a computer and a camera. The computer, equipped with an AMD Ryzen 7 4800H processor, an NVIDIA GeForce GTX 1650 graphics card, and 16 GB of RAM, will serve as the powerhouse. It will execute code and facilitate model training and testing, ensuring the model's robustness. Simultaneously, the camera will play a pivotal role in validating the object detection model, enabling real-time monitoring by seamlessly integrating with the computer's Windows 10 operating system.

2.2 Software
2.2.1 YOLOv8

The most recent iteration of the YOLO object detection model is called YOLOv8. It is depicted in Figure 1. This most recent version of YOLO shares the same architecture as its predecessors 6, but it makes many improvements over those found in the earlier iterations, including a new neural network architecture that makes use of both the Feature Pyramid Network (FPN) and Path Aggregation Network (PAN), as well as a new labeling tool that makes the annotation process easier. Numerous helpful features, like automatic labeling, labeling shortcuts, and programmable hotkeys, are included in this labeling tool.
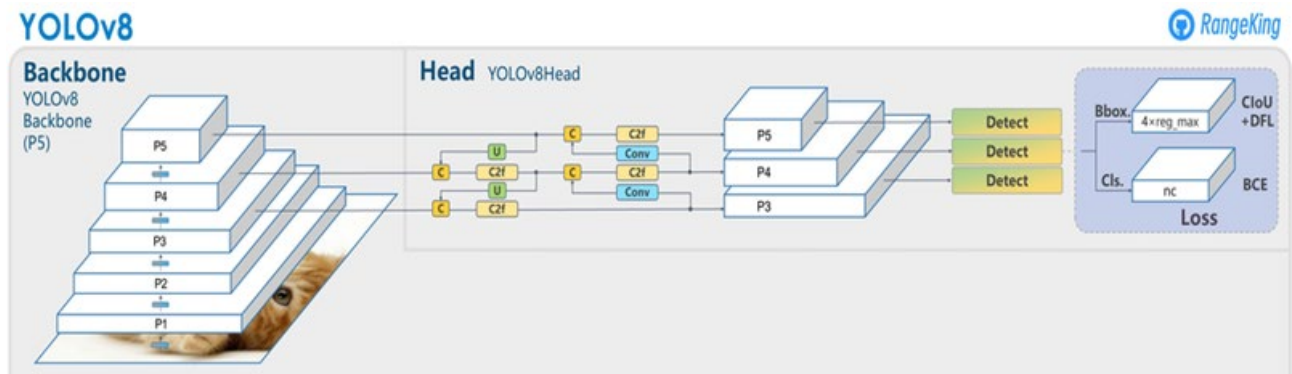
Fig. 1. YOLOv8 Architecture

2.2.2 Roboflow

Roboflow, a versatile cloud-based platform, offers an array of features that are instrumental in the annotation phase of the project. It allows the images to be annotated by a group of people using multiple devices rather than one single device, meaning the work can be split among multiple individuals (Figure 2). It also keeps track of the number of annotations, the class balances, image sizes, and provides annotation heatmaps, all to ensure the user knows what their datasets are doing [23].
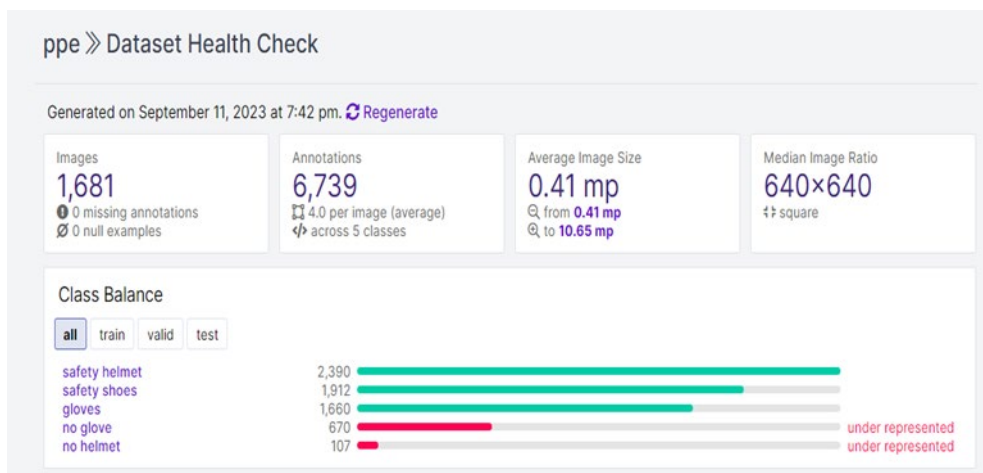

Fig. 2. The health check capabilities

Furthermore, Roboflow has the capability to create multiple versions of augmented images, ensuring that the original dataset remains intact, as augmentation is performed at the final step before the dataset is used for training (Figure 3). Additionally, Roboflow can modify an entire class's name during the pre-processing phase, allowing users to either exclude it completely or change the name to suit their needs. In the context of this project, multiple annotators were employed, health checks were used to monitor annotations, and pre-processing tools were utilized to adjust classes as part of feature selection to fine-tune the dataset for training [24].

Another part of Roboflow that was crucial to the project's methodology is its ability to export datasets through download codes. Normally, if a user uses cloud-based annotation tools, they will have to download the finished annotation dataset to run the training on the user's device. However, Roboflow allows the exporting of datasets to be done all in the cloud when paired with Google Colab. Just by copying the download codes and pasting them into Google Colab, the user can have the

dataset in Google Colab without the hassle of downloading it into the user's storage or downloading and reuploading it to Google Drive.
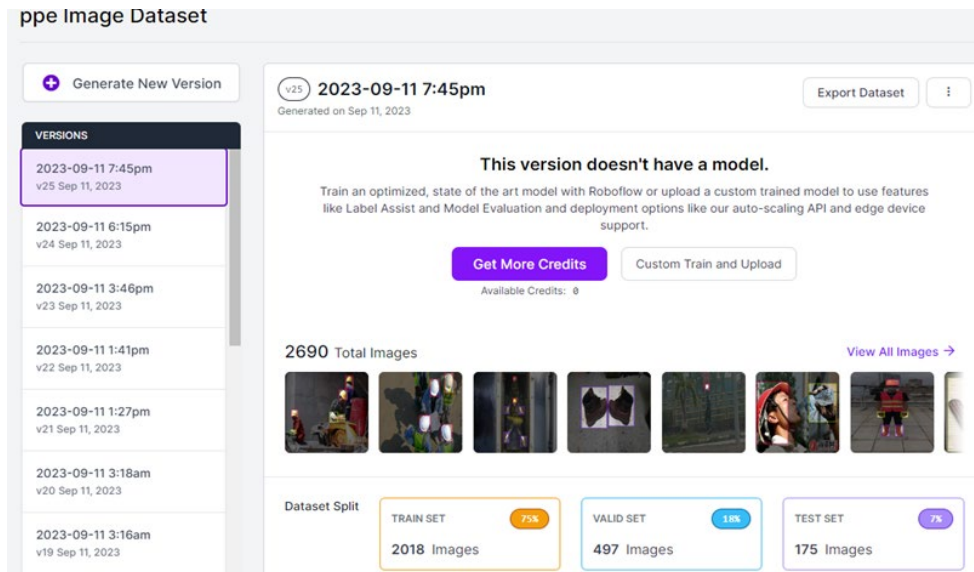

Fig. 3. Roboflow ability to retain past version of that dataset

This allows the user to adjust the dataset in a fast and efficient manner, allowing for the process of trial and error to finally get an agreeable result. In the matter involving the project, many trials and errors had to be done to achieve the expected results, and Roboflow's ability to allow for this kind of process to be done was crucial.

2.2.3 Google colaboratory

Google Colab is software that helps developers create, execute, and share Python code within a document using any browser. It's a variation of the well-known Jupyter Notebook and is included in the Google toolkit. Users can build a document using Jupyter Notebooks (and subsequently Google Colab) that incorporates executable code along with text, images, HTML, LaTeX, and other elements (Figure 4). The document is then saved to Google Drive and can be shared with peers and colleagues for editing, commenting, and viewing (Functions, n.d.).

One of the standout features of Google Colab is its provision of free access to Graphics Processing Units (GPUs) and Tensor Processing Units (TPUs). This is especially valuable for training computationally intensive deep learning models such as YOLOv8. Utilizing these hardware accelerators significantly speeds up model training compared to running it on a CPU. As Google Colab operates in the cloud, users are relieved of hardware constraints and the need to set up dependencies. It can be accessed from any device with an internet connection and a browser. Moreover, Google Colab comes pre-installed with many popular Python libraries and frameworks used in machine learning, including TensorFlow, PyTorch, and OpenCV, simplifying the setup process for YOLOv8 training.

In the context of the project, Google Colab's cloud-based software enables the direct upload of annotated datasets from Roboflow using download codes generated by Roboflow and loaded through its plugin. An example demonstrating how Roboflow download codes work in Google Colab is provided below (J. Glenn, "Ultralytics YOLOv8" [online]).

```
[ ]  !mkdir {HOME}/datasets
     !cd {HOME}/datasets

     !pip install roboflow

     from roboflow import Roboflow
     rf = Roboflow(api_key="ZGNmC00kCgGhgS5OYsjs")
     project = rf.workspace("test-on9ox").project("ppe-8uccm")
     dataset = project.version(25).download("yolov8")
```

Fig. 4. Example of how the download codes are put

2.3 Project Flow

The flowchart in Figure 5 illustrates the sequential steps in the project, highlighting the integration of Roboflow and Google Colab as key components. The process starts with data annotation and preprocessing in Roboflow, then it transfers the datasets to Google Colab. Within Google Colab, code development and model training take place, leveraging the platform's cloud-based infrastructure and hardware accelerators. The collaborative aspect of Google Colab enables easy sharing and interaction with peers. This well-defined flowchart visually represents the project's streamlined and efficient workflow.
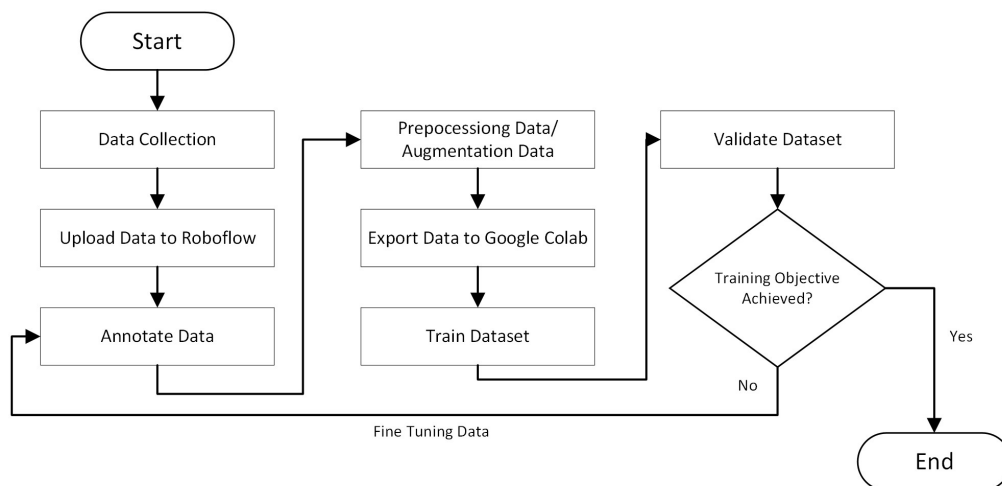
Fig. 5. Flow chart of the project

2.3.1 Data collection

Data collection is done through manual collection as well as downloading datasets from the internet and other sources. Landscape and portrait pictures are used and will be resized to 640x640 using Roboflow's pre-processing options. Landscape shots are used when taking pictures of multiple individuals, usually in the case of taking pictures of workers in PPE walking around the plantation, which is mostly far away. While portraits are used in shots involving closeups to allow more variation,

2.3.2 Uploading data to roboflow

When starting to upload the data, a file selection dialog will appear, allowing users to browse their local computer's storage to choose the images to include in the dataset. Multiple images can be selected at once by holding down the Ctrl key (or Command key on Mac) while clicking on the desired files. Alternatively, multiple image files can be dragged and dropped directly into the upload area.

Roboflow also supports various image formats, including common formats like JPEG and PNG, meaning multiple different sources can be used without problems. It will also gather all the data into batches. Subsequent uploads will be divided into batches to allow users to keep track of when and how many uploads they do in a batch. Once all the images are annotated, they will be moved to one large dataset, but in the annotated section, the images will still be labeled in batches, which allows users to delete them without having to manually search for them in case the data is bad. Which did happen a few times in the project.

2.3.3 Annotate data

During the annotation process, once the box tool is clicked and the object is labeled, a prompt will appear where the class name can be specified, or if a class already exists, the label can then be named after the pre-existing class. On the left, it shows how many classes are in a picture, and below the class table, it shows the unused classes that are not present for this image. The classes are also labeled with different colors to help differentiate the data. If, during the validation process, the results are shown to be insufficient, then the images will go through a fine-tuning process.

Where either more data is added or feature selection is conducted. In feature selection, more classes are added to reduce false positives; for example, some of the datasets have people not wearing gloves. Because of the similar shape, the machine learning might mistake hand-looking objects for gloves, so the class "No Gloves" is added to focus on what is a glove and what is not. That includes helmets; some of the datasets do not have people wearing helmets, so "No Helmets" is added. Other than that, problems in the results might stem from a lack of data or overfitting the model with similar data; thus, some data will be removed or added, or the boxes need to fill the object more tightly.

2.3.4 Pre-processing data/augmentation data

After annotation, the dataset can then be processed to generate a working version of it. This creates another separate dataset that will be used for training. This dataset can be run through pre-processing that has multiple options, such as auto-orient, which will make any rotated images straight; resize if any of the images have different dimensions; and more, such as grayscale, if the user wants to train images to learn shapes instead of colors.

Another process is augmentation, a process that creates a copy of an image in the dataset but one that is slightly different from it depending on the options chosen. For example, several options were used to train during the project, such as Flip. Which flips the images so that later in the training, the model will learn different orientations. Then there is rotation that rotates the image to a specific degree, grayscale to ensure that the model will be trained to see an object not just by the colors but by the shape, brightness so that it will be trained in bright and dark backgrounds, and finally hue to change the colors to different colors.

2.3.5 Export data to google colab

After generating a version of the dataset that includes augmentation and pre-processing, the dataset can then be exported to Google Colab through the use of a download code that can be generated into multiple formats that include XML, TXT, JSON, CSV, and more. Then simply paste the code into a cell in Google Colab and run it. Pytorch modules are required to be installed as they did not come installed in Colab; some do, but not all [25]. Once it runs, it'll download the dataset into

temporary storage in Google Colab. Do not forget that once the runtime expires, the data will be terminated and lost, but the data can be stored into the user's Google drive through another process that will be explained at the end of the flowchart.

2.3.6 Training dataset

Before the training, the runtime was changed to GPU or TPU for the hardware accelerator to improve training speed. Training it with a CPU would have taken hours. Test and regret: 9 hours for 40 epochs. Some of the options that could be added were the learning rate, which by default is 0.001. The project has tested a learning rate of 0.001, 0.01, 0.05, 0.1, and 0.5. Google Drive can be mounted in Colab for easy access to datasets and model files; however, for this project, mounting the Google Drive is done to store model files and results instead of taking them from there. If the training results prove to be insufficient and fine-tuning the data is required, one simply has to copy a new version of a better dataset download code and paste it into an earlier cell, preferably one above the cell containing the training code. No extensive recoding is required when retraining the data. However, because Google Colab is cloud-based software, if the runtime is terminated, any unsaved data will be deleted, so it is important to download any important data before hand or have backups in the cases of video prediction work where it is required to upload a video to Google Colab for it to validate.

2.3.7 Validate dataset

When validating the dataset, several performance metrics were used, such as mean average precision (mAP), precision, and recall. The furthest on the right is mAP50–95, which is average mAP over different IoU thresholds, from 0.5 to 0.95. To further elaborate on each of the metrics, the first is the mean average precision, which is one of the most widely used metrics for evaluating object detection models like YOLOv8. It assesses the precision-recall trade-off across multiple object classes.

Precision measures the accuracy of positive predictions (how many are correct), while recall quantifies the model's ability to find all positive instances in the dataset. The mAP score is the average precision calculated for each class and then averaged across all classes. It provides an overall assessment of the model's detection accuracy. mAP values range from 0 to 1, so a value closer to 1 means it is good; thus, just because the value is 0.8 does not mean it is small; it means it is close to very good, which is a singular 1. To further validate the data, the model was also tested on a video [Roboflow] (2023, January 11). This is because previous versions of the dataset have given decent numbers but failed to reliably detect objects when seen in the prediction data. The data, however, needs to be short because during testing, videos will come out significantly larger than before. For example, a 16-MB video that was about 17 seconds long was used to validate the data. After it was finished and uploaded to Google Drive, it is now more than 10 times the size, around 274 MB. While it can take on larger and longer videos, it will take a significantly longer time to process, and during the process, it slightly shows what is being detected in some frames as well.

2.3.8 Training objectives achieved

The objective is to create a model that can detect safety attire reliably; if the results show that it cannot, then it must be retrained with a better dataset. An example of a bad result is shown below. In the example above, the model detected the surroundings as gloves and safety shoes. While the confidence level varies on the lower side, it persists for far too long to be considered reliable; thus, this must be remedied by either additional data or feature selection. Also, it failed to detect some of

the gloves, even though they were clearly visible. The example in Figure 6 is an acceptable result. Not only is there no background detection, but the individual class detection has high confidence levels.

(a)	(b)

Fig. 6. (a) Shows the outcome of insufficient results and (b) the outcome acceptable results (right)

3. Results
3.1 Feature Engineering

Feature engineering involves the process of enhancing the performance of predictive modeling on a given dataset through the manipulation and transformation of its feature space. The current approaches for automating this process involve either expanding the dataset explicitly with all transformed features and then performing feature selection, or exploring the transformed feature space through evaluation-guided search. We will utilize a method known as feature selection to identify and define two closely interconnected entities.

The feature selection that was chosen was the exclusion of helmets and gloves, as these variables are highly correlated with helmets and gloves (Figure 7). This aids the model in establishing the criteria for classifying an object as a glove or not a glove, and vice versa. The exclusion of safety shoes from the feature selection can be attributed to the highly improbable scenario of individuals engaging in work activities without any form of footwear.

Fig. 7. Example of feature selection

3.2 Optimal Result

The YOLOv8 model has been trained on the dataset using transfer learning, where we initialized the model with pretrained weights on the COCO dataset and fine-tuned it on our dataset [26]. The best model was saved as best.pt [27] when the training procedure was complete, and the model displayed its best performance at epoch 47. The training was then stopped at 50 epochs, as previous training showed that past 50 epochs, the training had stopped showing improvement. Because detecting safety attire requires multiple targets, it is a complex scene. By employing a feature selection technique by adding additional classes, this is to reduce overfitting as features that have a high correlation to each other can be differentiated, and while the number of these classes will be lightweight, it will ensure a more accurate reading [28]. Furthermore, refer to Table 1 for the outcome of the optimal run, Table 2 for the configuration and variables to obtain the optimal result, and Table 3 for the classes used for training and testing to obtain the optimal result. Based on the figures below, it can be observed that the learning readings for all the objects except those used for feature engineering were above 0.8. The safety helmet achieved 0.969, the safety gloves achieved 0.857, followed by the safety vest with 0.887, respectively.

Table 1
Outcome of the optimal run

Class	Images	Instances	Precision	Recall	mAP50	mAP50-95):
all	497	2097	0.79	0.756	0.815	0.527
gloves	497	554	0.84	0.8	0.857	0.591
no glove	497	184	0.606	0.516	0.542	0.251
no helmet	497	24	0.736	0.698	0.821	0.55
safety helmet	497	752	0.927	0.934	0.969	0.626
safety shoes	497	583	0.839	0.832	0.887	0.617

Table 2
Configuration and variables to obtain optimal result

Number of Class	5
Total Dataset	1681
Total Trained Dataset	70%
Total Tested Dataset	10%
Total Valid Dataset	2%
Annotations	6739

Table 3
Classes used for training and testing to obtain optimal result

Classes	Object	Number of Dataset
Class 1	Safety Helmet	2390
Class 2	Safety shoes	1912
Class 3	Gloves	1660
Class 4	No Gloves	670
Class 5	No Helmets	107

Figure 8 shows the results in graph form. An epoch represents a complete iteration through the entire image dataset in YOLO. During each epoch, the YOLO model traverses all the bounding boxes in the dataset and updates its model parameters based on the loss function and optimization algorithm mentioned above. The higher the number of epochs, the more times the YOLO model will go through the entire dataset. However, caution must be exercised while selecting the epoch number. A low epoch number might cause underfitting when crucial visual characteristics are missed

by the model. On the other hand, a large epoch number might lead to overfitting, in which the model exhibits an excessive bias towards the training set and performs badly on unrelated datasets. As a result, picking a suitable epoch number frequently requires trial and error. [29]
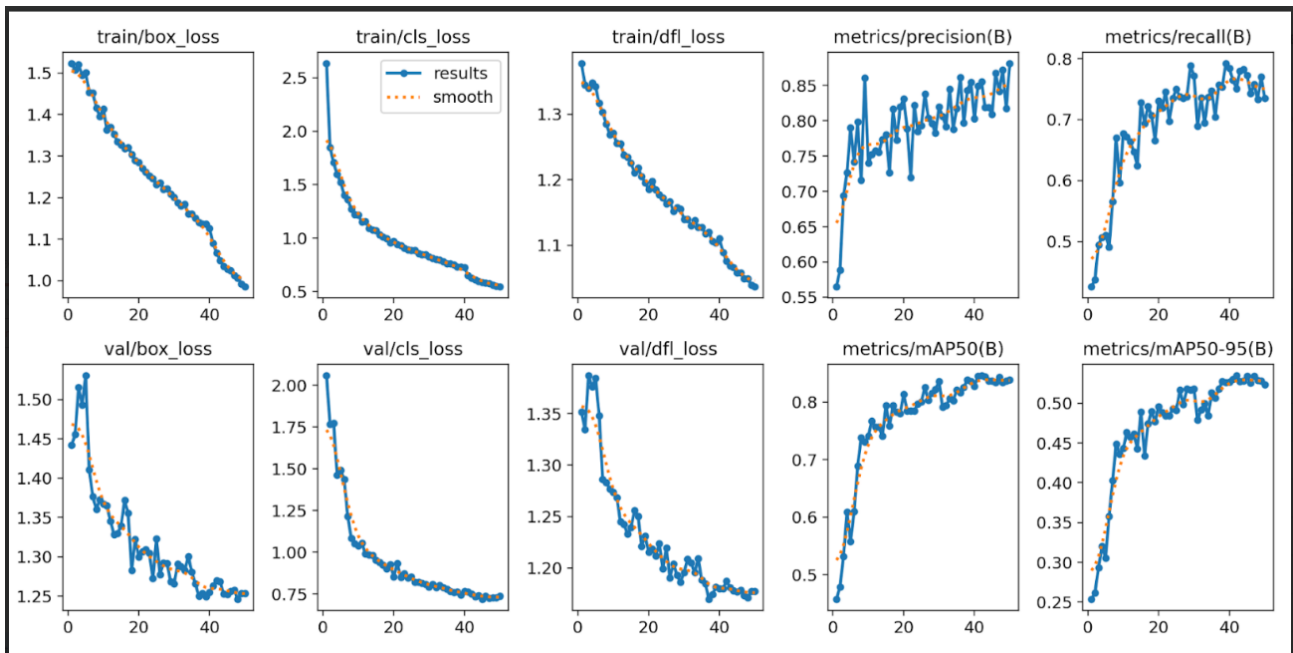

Fig. 8. The matrix result of the training

Another noteworthy observation is the spikes in the box_loss graph; it is still showing signs of reducing patterns. A smooth curve instead shows that the data is trained easily and is prone to overfitting. Another is cls_loss. It is observed that in the initial stages of val/cls_loss, the model is still adapting to the validation data, and its predictions might vary. Once the model has adapted, the lines begin to smooth out. Then, in Figure 9, the confusion matrix is plotted for true and the five predicted classes. The introduction of feature learning to the classes has helped reduce the false positives between the classes by adding features to help differentiate between the three main classes. Figure 10 indicates that YOLO v8 has obtained an average F1-score of 0.8 for all classes. Likewise, the YOLO v8 model has obtained better recall and precision of 0.93 and 1, respectively, for all classes. This proves the effectiveness of YOLO v8 for object detection.
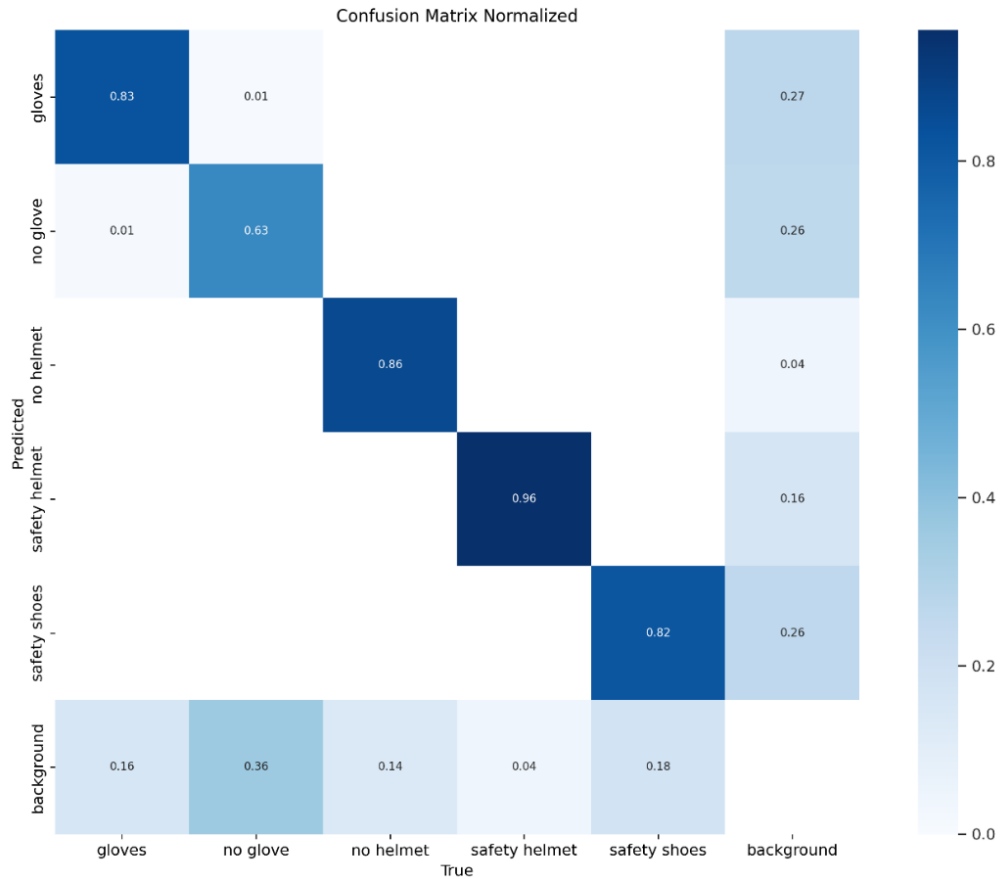
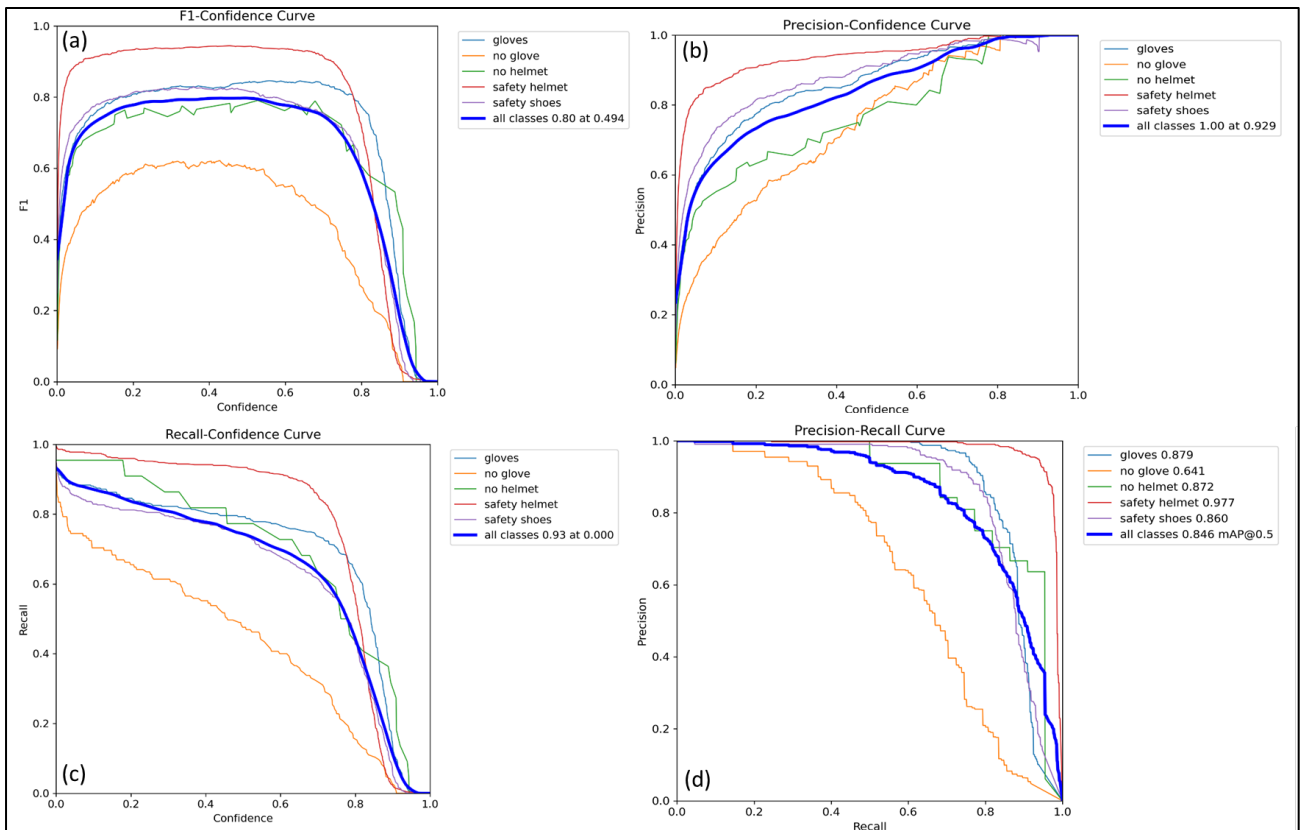**Fig. 9.** The confusion matrix for optimal result

Fig. 10. (a) F1 Confidnce graph for optimal result (b) Precision confidence graph for optimal result (c) Recall-Condidence graph for optimal result (d) precision recall graph for optimal result

3.3 Result with no Implementation of Feature Selection

The Table 4 and Figure 11 below shows the results obtained without the use of feature engineering. While the numbers are higher in most metrics, the recall is slightly lower than the one with additional classes. This is most likely because the pixels of small targets are small, which may cause objects to overlap, especially in more complex scenes, making it more difficult to recall information [28].

Table 4
Outcome of training without feature selection

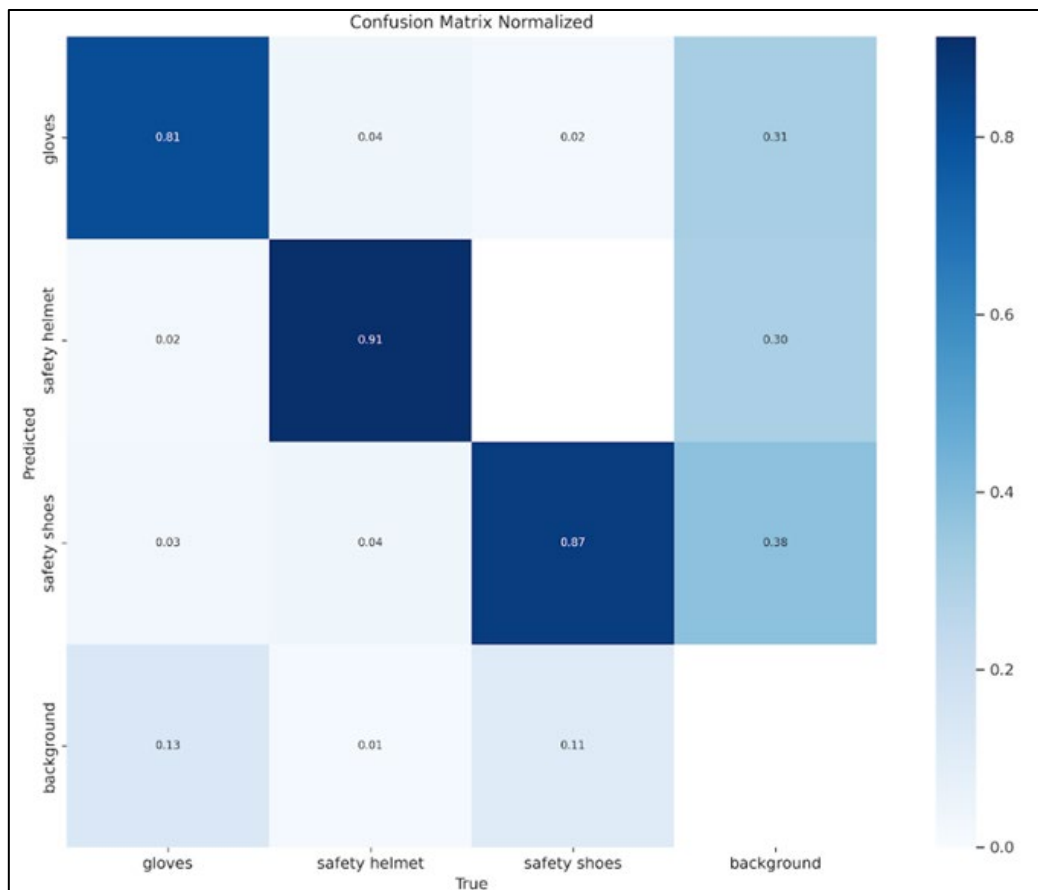| Class	Images	Instances	Precision	Recall	mAP50	mAP50-95):
all	497	1160	0.922	0.827	0.902	0.606
gloves	497	554	0923	0.799	0.876	0.626
safety helmet	497	752	0.932	0.939	0.977	0.624
safety shoes	497	583	0.911	0.744	0.853	0.568

Fig. 11. The confusion matrix of the model without feature selection

While the results are good, it might be false positives that are causing the higher numbers. Adding additional classes also helped make the model more flexible and detect when a worker is not wearing a helmet or gloves. As seen in Figure 12, it is a comparison result between 3 classes and 5 classes. Because the dataset is trained with heads, it detects nearly all heads as safety helmets, but by adding an additional class, it is now trained to differentiate between no helmets and helmets and also reduce irrelevant and redundant features from the dataset [30]. Tests were done to instead add Face as a class, but it did not help in differentiating between faces with no helmet and helmets, most likely

because the Yolov8 model has already been trained for faces, which is why despite the results giving a high recall and precision rate for Face, it didn't detect.
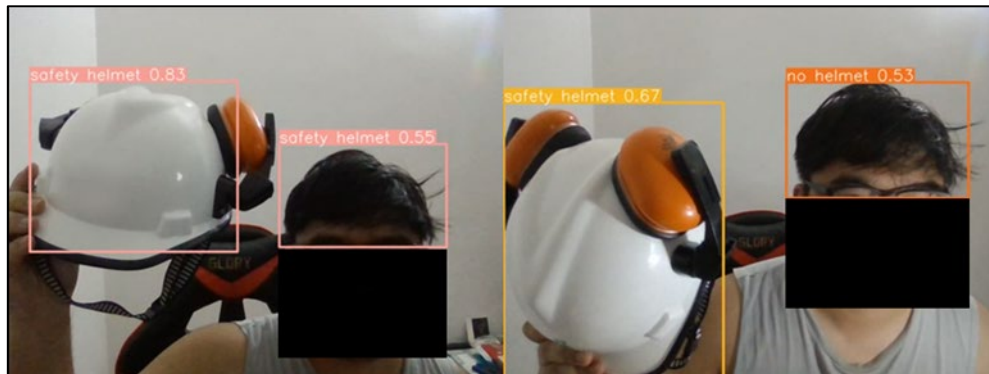

Fig. 12. Live feed comparison between model with additional class and model without.

4. Conclusions

In conclusion, this study aimed to delve into the capabilities of YOLOv8, an enhanced iteration of the potent object detection model. Despite its initial limitation of being trained on higher-resolution models, it has shown exceptional promise in safety attire detection. Building upon the foundation of occupational accident statistics, our research sought to unlock the potential of machine learning to enhance safety by ensuring the proper use of personal protective equipment (PPE). This project proposed the development of an object detection model using YOLOv8 to identify safety attire, including gloves, no gloves, safety helmets, no safety helmets, and safety shoes. We meticulously curated a dataset consisting of 1681 annotated images, which was subsequently divided into training, validation, and testing sets at a ratio of 70%, 20%, and 10%, respectively. Throughout this project, we undertook multiple iterations and meticulously fine-tuned the configuration of the object detection model. We also explored the impact of varying the number of classes and the number of images in each class on the model's performance. In cases where we encountered objects with similar physical structures, like gloves and hands, we implemented feature selection to eliminate redundancy and irrelevant information from the training process, thereby improving the model's precision and efficiency. This study not only sheds light on the power of YOLOv8 but also emphasizes its practicality in real-world applications, particularly in ensuring workplace safety through automated PPE compliance monitoring. Our findings contribute to the ongoing dialogue on enhancing safety measures, and we are committed to furthering the understanding and implementation of advanced object detection models in various industries.

Acknowledgement

References

[1] Wieczorowski, Michal, Dawid Kucharski, Pawel Sniatala, Pawel Pawlus, Grzegorz Krolczyk, and Bartosz Gapinski. "A novel approach to using artificial intelligence in coordinate metrology including nano scale." Measurement 217 (2023): 113051. https://doi.org/10.1016/j.measurement.2023.113051
[2] Milligan, Ian. The transformation of historical research in the digital age. Cambridge University Press, 2022. https://doi.org/10.1017/9781009026055

[3] Hindocha, S., K. Zucker, R. Jena, Kathryn Banfill, K. Mackay, G. Price, D. Pudney, J. Wang, and A. Taylor. "Artificial intelligence for radiotherapy auto-contouring: current use, perceptions of and barriers to implementation." Clinical Oncology 35, no. 4 (2023): 219-226. https://doi.org/10.1016/j.clon.2023.01.014

[4] Sazali, N., W. N. W. Salleh, A. F. Ismail, N. H. Ismail, F. Aziz, N. Yusof, and H. Hasbullah. "Effect of stabilization temperature during pyrolysis process of P84 co-polyimide-based tubular carbon membrane for H2/N2 and He/N2 separations." In IOP Conference Series: Materials Science and Engineering, vol. 342, no. 1, p. 012027. IOP Publishing, 2018. https://doi.org/10.1088/1757-899X/342/1/012027

[5] Sharma, Neha, Reecha Sharma, and Neeru Jindal. "Machine learning and deep learning applications-a vision." Global Transitions Proceedings 2, no. 1 (2021): 24-28. https://doi.org/10.1016/j.gltp.2021.01.004

[6] Alzubaidi, Laith, Jinglan Zhang, Amjad J. Humaidi, Ayad Al-Dujaili, Ye Duan, Omran Al-Shamma, José Santamaría, Mohammed A. Fadhel, Muthana Al-Amidie, and Laith Farhan. "Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions." Journal of big Data 8 (2021): 1-74. https://doi.org/10.1186/s40537-021-00444-8

[7] Möller, Dietmar PF. "Machine Learning and Deep Learning." In Guide to Cybersecurity in Digital Transformation: Trends, Methods, Technologies, Applications and Best Practices, pp. 347-384. Cham: Springer Nature Switzerland, 2023. https://doi.org/10.1007/978-3-031-26845-8_8

[8] Haleem, Abid, Mohd Javaid, Mohd Asim Qadri, and Rajiv Suman. "Understanding the role of digital technologies in education: A review." *Sustainable Operations and Computers* 3 (2022): 275-285. https://doi.org/10.1016/j.susoc.2022.05.004

[9] Taye, Mohammad Mustafa. "Understanding of Machine Learning with Deep Learning: Architectures, Workflow, Applications and Future Directions." *Computers* 12, no. 5 (2023): 91. https://doi.org/10.3390/computers12050091

[10] Ding, Sheng, and Kun Zhao. "Research on daily objects detection based on deep neural network." In *IOP conference series: materials science and engineering*, vol. 322, no. 6, p. 062024. IOP Publishing, 2018. https://doi.org/10.1088/1757-899X/322/6/062024

[11] Cao, Danyang, Zhixin Chen, and Lei Gao. "An improved object detection algorithm based on multi-scaled and deformable convolutional neural networks." *Human-centric Computing and Information Sciences* 10, no. 1 (2020): 1-22. https://doi.org/10.1186/s13673-020-00219-9

[12] Pathak, Ajeet Ram, Manjusha Pandey, and Siddharth Rautaray. "Application of deep learning for object detection." *Procedia computer science* 132 (2018): 1706-1717. https://doi.org/10.1016/j.procs.2018.05.144

[13] Delhi, Venkata Santosh Kumar, R. Sankarlal, and Albert Thomas. "Detection of personal protective equipment (PPE) compliance on construction site using computer vision based deep learning techniques." *Frontiers in Built Environment* 6 (2020): 136. https://doi.org/10.3389/fbuil.2020.00136

[14] Lo, Jye-Hwang, Lee-Kuo Lin, and Chu-Chun Hung. "Real-Time Personal Protective Equipment Compliance Detection Based on Deep Learning Algorithm." *Sustainability* 15, no. 1 (2022): 391. https://doi.org/10.3390/su15010391

[15] Nath, Nipun D., Amir H. Behzadan, and Stephanie G. Paal. "Deep learning for site safety: Real-time detection of personal protective equipment." *Automation in Construction* 112 (2020): 103085. https://doi.org/10.1016/j.autcon.2020.103085

[16] Li, Guojin, Xiaojie Huang, Jiaoyan Ai, Zeren Yi, and Wei Xie. "Lemon-YOLO: An efficient object detection method for lemons in the natural environment." *IET Image Processing* 15, no. 9 (2021): 1998-2009. https://doi.org/10.1049/ipr2.12171

[17] De Carolis, Berardina, Francesco Ladogana, and Nicola Macchiarulo. "Yolo trashnet: Garbage detection in video streams." In *2020 IEEE Conference on Evolving and Adaptive Intelligent Systems (EAIS)*, pp. 1-7. IEEE, 2020. https://doi.org/10.1109/EAIS48028.2020.9122693

[18] Liu, Wenyu, Gaofeng Ren, Runsheng Yu, Shi Guo, Jianke Zhu, and Lei Zhang. "Image-adaptive YOLO for object detection in adverse weather conditions." In *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 2, pp. 1792-1800. 2022. https://doi.org/10.1609/aaai.v36i2.20072

[19] Hu, Xuelong, Yang Liu, Zhengxi Zhao, Jintao Liu, Xinting Yang, Chuanheng Sun, Shuhan Chen, Bin Li, and Chao Zhou. "Real-time detection of uneaten feed pellets in underwater images for aquaculture using an improved YOLO-V4 network." *Computers and Electronics in Agriculture* 185 (2021): 106135. https://doi.org/10.1016/j.compag.2021.106135

[20] Cheung, Weng-Fong, Tzu-Hsuan Lin, and Yu-Cheng Lin. "A real-time construction safety monitoring system for hazardous gas integrating wireless sensor network and building information modeling technologies." *Sensors* 18, no. 2 (2018): 436. https://doi.org/10.3390/s18020436

[21] Arthur-Holmes, Francis, and Kwaku Abrefa Busia. "Safety concerns and occupational health hazards of women in artisanal and small-scale mining in Ghana." *The Extractive Industries and Society* 10 (2022): 101079. https://doi.org/10.1016/j.exis.2022.101079

[22] Ricci, Federico, and Massimo Nucci. "Safety Integrated Model-training (SIM-t) and its evaluation: A safety training proposal for mechanical companies." *Safety science* 146 (2022): 105538. https://doi.org/10.1016/j.ssci.2021.105538

[23] Ibrahim, Haziqatulhanis, Norazlianie Sazali, Ahmad Syahiman Mohd Shah, Mohamad Shaiful Abdul Karim, Farhana Aziz, and Wan Norharyati Wan Salleh. "A review on factors affecting heat transfer efficiency of nanofluids for application in plate heat exchanger." *Journal of Advanced Research in Fluid Mechanics and Thermal Sciences* 60, no. 1 (2019): 144-154.

[24] Sari, Marlynda Happy Nurmalita, Dina Dewi Anggraini, and Yuli Kusumawati. "The Effectiveness of Pregnancy Online Classes (PROCLASS) on the Level of Knowledge and Anxiety Ahead of Labor During the COVID-19 Pandemic." Jurnal Kebidanan dan Kesehatan Tradisional (2023): 10-19. https://doi.org/10.37341/jkkt.v8i1.410

[25] Panchal, Ronak. "A Study of Real World Data Visualization of COVID-19 dataset using Python." International Journal of Management and Humanities 4, no. 8 (2020): 104-107. https://doi.org/10.35940/ijmh.H0834.044820

[26] Talaat, Fatma M., and Hanaa ZainEldin. "An improved fire detection approach based on YOLO-v8 for smart cities." *Neural Computing and Applications* 35, no. 28 (2023): 20939-20954. https://doi.org/10.1007/s00521-023-08809-1

[27] Yang, Jiesheng, Andreas Wilde, Karsten Menzel, Md Zubair Sheikh, and Boris Kuznetsov. "Computer Vision for Construction Progress Monitoring: A Real-Time Object Detection Approach." *arXiv preprint arXiv:2305.15097* (2023). https://doi.org/10.1007/978-3-031-42622-3_47

[28] Lou, Haitong, Xuehu Duan, Junmei Guo, Haiying Liu, Jason Gu, Lingyun Bi, and Haonan Chen. "DC-YOLOv8: Small-Size Object Detection Algorithm Based on Camera Sensor." *Electronics* 12, no. 10 (2023): 2323. https://doi.org/10.3390/electronics12102323

[29] Soylu, Emel, and Tuncay Soylu. "A performance comparison of YOLOv8 models for traffic sign detection in the Robotaxi-full scale autonomous vehicle competition." *Multimedia Tools and Applications* (2023): 1-31. https://doi.org/10.1007/s11042-023-16451-1

[30] Boonsirisumpun, Narong, Wichai Puarungroj, and Phonratichi Wairotchanaphuttha. "Automatic detector for bikers with no helmet using deep learning." In 2018 22nd International Computer Science and Engineering Conference (ICSEC), pp. 1-4. IEEE, 2018. https://doi.org/10.1109/ICSEC.2018.8712778