FINAL REPORT

Name	ID
Elsayed Sofy Elsayed	2400288

Data

Roboflow	**https://app.roboflow.com/elsayed-sofy-ulmwo/final-progect/1**
Google Drive	**https://drive.google.com/drive/folders/18zRIYw8_BbEj3eBXv99GHVPrMyrwd6bj?usp=drive_link**
contact	**Sayedsofy181232@gmail.com**

1)INTRODUCTION

Technology has greatly influenced human history . Deep learning and machine learning have recently emerged as transformative fields, significantly advancing our comprehension of computer capabilities. The primary goal of this objective is to equip machines with the inherent human capacity for perception, reasoning, and practical application of knowledge . Object detection is a crucial aspect of computer vision that has various applications in different industries, particularly in the context of ongoing technological advancements . One important application of personal protective equipment (PPE) is to ensure safety and compliance in various environments . 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 . 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 . 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 . Challenges arise when individuals disregard safety regulations and work without appropriate attire, especially when they go unnoticed by health and safety officials . 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)Literature Review

PPE serves the vital purpose of safeguarding its wearer's body from occupational hazards and preventing accidents. However, several factors, including low awareness of PPE usage, discomfort, fatigue, and negligence, contribute to insufficient PPE utilization and incorrect handling among workers . Research conducted by concerning accurate PPE usage detection at construction sites has highlighted the potential of computer–vision based methods for automatically detecting PPE completeness. These methods offer non invasive, cost-effective perception on-site, as they typically identify all workers and PPE components before verifying if a worker is using PPE based on their relationship with the involved equipment. The need to detect PPE arises in order to enhance the completeness and accuracy of its usage, thereby reducing workplace accidents . Consequently, there exists a significant practical requirement to utilize technology that can assist practitioners in improving or ensuring PPE completeness. While efforts to incorporate electronic circuits into PPE have been made , technologies allowing visual and noncontact completeness adherence to safety regulations are more prevalent and practical to implement. To accommodate this requirement, deep learning becomes a prominent approach for PPE detection. In research by , a CNN was used to concurrently detect workers' use of hard hats and vests. Overall, CNN-based methods directly process images of workers and classify the status of PPE usage, including both the level of completeness and accuracy, through an end-to-end inference process. A research conducted research on PPE detection with the aim of reducing workplace accidents in the construction industry . Their study employed a CNN to detect PPE usage by workers and classify various types of PPE, such as determining if each worker wears a hard hat. Beyond hard hats, some studies have extended PPE detection to various tools, with simultaneous detection processes. Ref. detected multiple types of PPE, including hard hats and vests, using a CNN for a comprehensive safety assessment. Moreover, Ref. used a CNN to simultaneously detect workers' use of hard hats and vests. Overall, CNN-based methods directly process images of workers, classifying both the level of completeness and accuracy of PPE usage. Earlier research on PPE completeness and accuracy detection using deep learning extensively analyzed and reviewed various deep learning algorithms employed in devel oping systems aimed at identifying PPE usage. The selection of these algorithms is based on the methods or techniques used in the developed systems to identify the presence of PPE objects. In a study conducted by which focused on PPE detection at construction sites, the YOLO detection method was used to identify hard-hat-wearing personnel. The detection of PPE objects was carried out using YOLOv3 and YOLOv4. In the PPE detection process, the proposed model offered practical detection performance in terms of speed and accuracy. This method holds significant potential for automated inspection of PPE components. Based on testing results, it was able to achieve detection efficiency of over 25 FPS and amAP value of 97%, which can be utilized to ascertain whether construction per-sonnel adhere to safety regulations and meet real-time, high-accuracy requirements. This demonstrates that YOLOv3 possesses high accuracy and detection speed, while YOLOv4 outperforms YOLOv3, particularly in terms of detecting small objects with improved speed and accuracy. The YOLO algorithm exists in multiple versions, each with its own performance char acteristics. In a study conducted by, the focus was on PPE detection at construction sites, specifically considering the use of YOLOv5. Their study aimed to detect PPE usage among

construction workers across six PPE categories: shoes, jackets, vests, gloves, glasses, and hard hats. The performance of the proposed YOLOv5s model variant was compared to other algorithms through three indicators: precision, recall, and F1 score. Comparative al gorithms included YOLOv4, Faster-RCNN MobileNetV3, and Faster-RCNN Resnet50. The results of a five-fold cross-validation technique revealed that YOLOv5s exhibited the most effective performance in terms of precision and recall indicators. The enhanced YOLOv5 model yielded the highest precision and recall values compared to benchmark models.

3)Methodology

3.1 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.

3.2 Software

3.2.1 YOLOv5,YOLOv7 and 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.

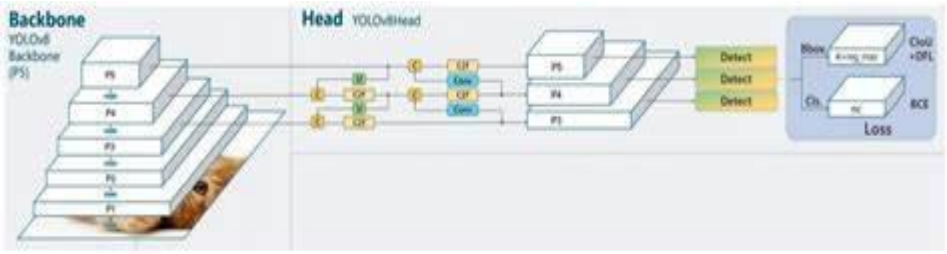

Fig. 1. YOLOv8 Architecture

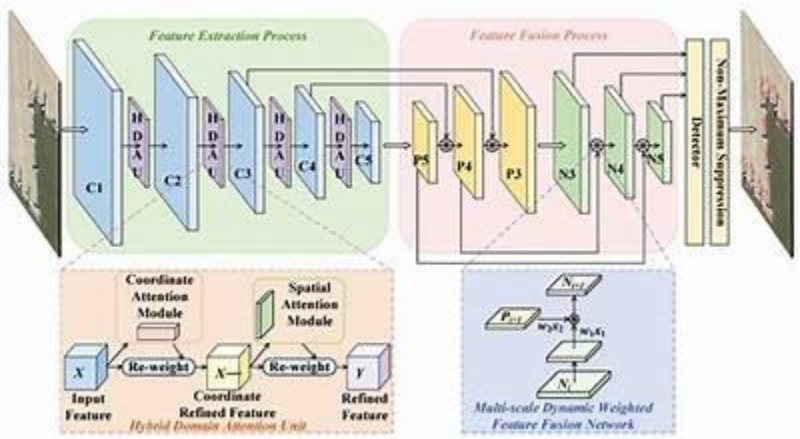

Fig. 2. YOLOv5 Architecture

3.2.2 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 3). 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

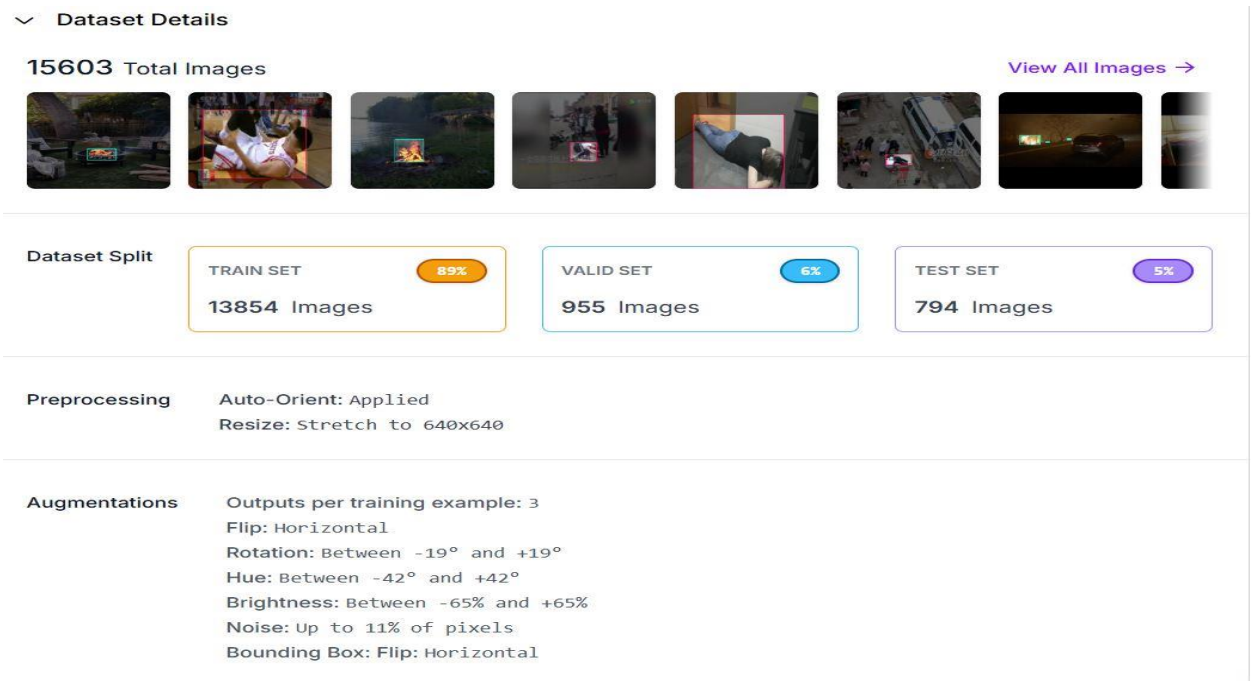

Fig. 3. Roboflow ability to retain past version of that dataset

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 4). 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 . 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. 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.

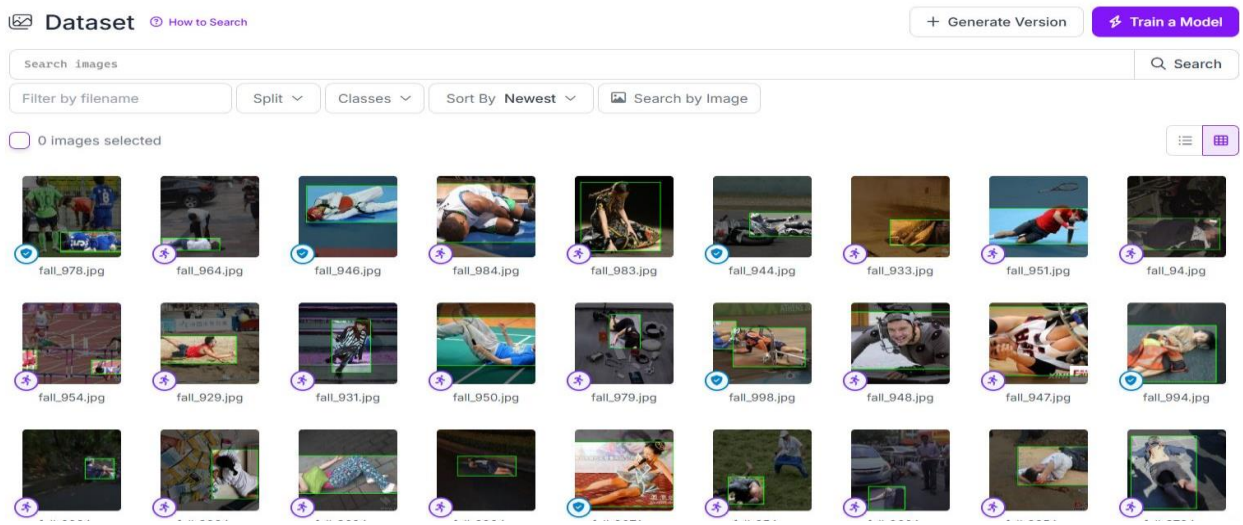

Fig.4. the dataset is used for training

3.2.3 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 5). 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 YOLOv5. 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 YOLOv5 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 YOLOv5” [online]).

```
!pip install roboflow

from roboflow import Roboflow
rf = Roboflow(api_key="foZEISZbHwud9lYPYdxU")
project = rf.workspace("elsayed-sofy-ulmwo").project("final-progect")
version = project.version(1)
dataset = version.download("yolov5-obb")
```

Fig. 5. Example of how the download codes are put

3.3 Project Flow

The flowchart in Figure 6 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.

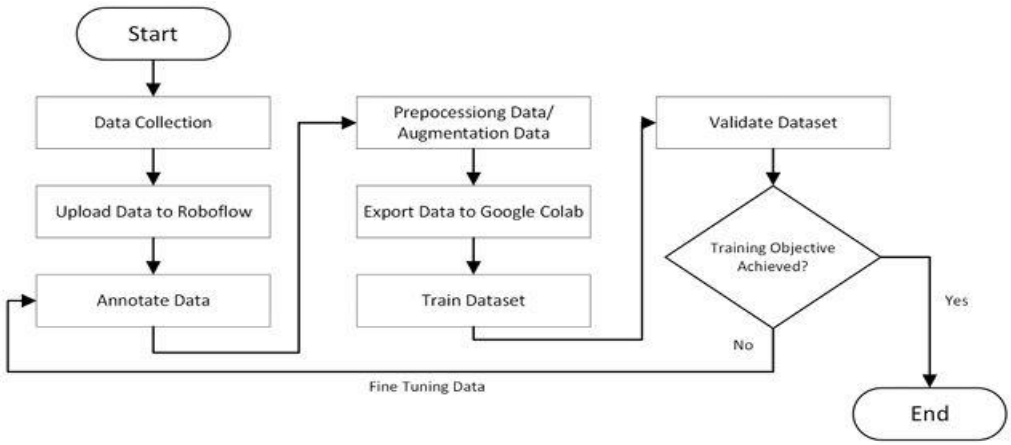

Fig. 6. Flow chart of the project

3.3.1 Data collection

Data is collected through Roboflow as well as datasets downloaded from the internet and other sources. Landscape and portrait images are used and will be resized to 640 x 640 using Roboflow's pre-processing options. Landscape shots are used when capturing multiple individuals, typically when capturing workers in PPE walking around a farm, which is often remote. Portraits are used for shots involving close-ups to allow for more variety.

3.3.2 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.

3.3.3 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.

3.3.4 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.

3.3.5 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 . 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.

3.3.6 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 150 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.

4. Results

4.1 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

4.2 Optimal Result

The YOLOv5 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 . The best model was saved as best.pt when the training procedure was complete, and the model displayed its best performance at epoch 127. The training was then stopped at 150 epochs, as previous training showed that past 150 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 . 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.

Class	Images	Precision	Recall	mAP50	mAP50-95
all	**6367**	**0.789**	**0.63**	**0.66**	**0.527**
gloves	**618**	**0.84**	**0.8**	0.857	0.591
No glove	**964**	0.606	0.516	0.542	0.251

No helmet	**1119**	0.736	0.698	0.821	0.55
safety helmet	**402**	0.927	0.934	0.969	0.626
safety shoes	**462**	0.839	0.832	0.887	0.617
fire	**4672**	**0.985**	**0.693**	**0.60**	**0.846**

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.

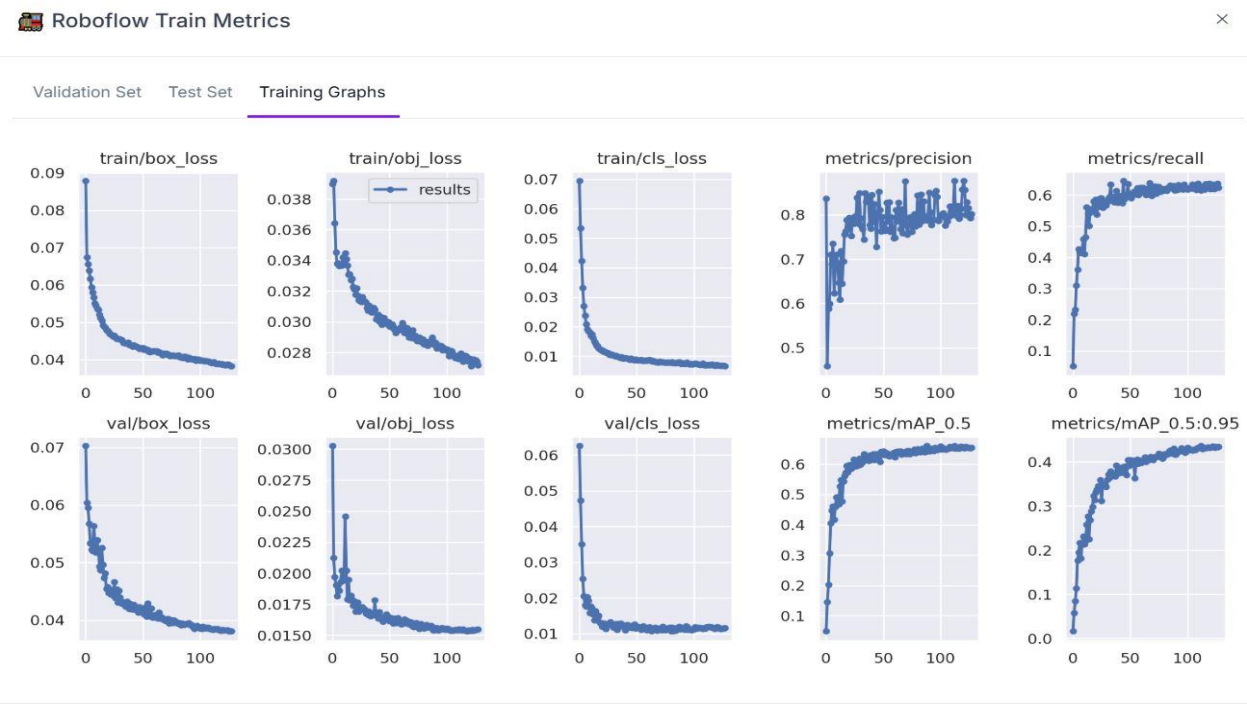

Fig. 8. The matrix result of the training

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 v5 has obtained an average F1-score of 0.8 for all classes. Likewise, the YOLO v5 model has obtained better recall and precision of 0.93 and 1, respectively, for all classes. This proves the effectiveness of YOLO v5 for object detection.

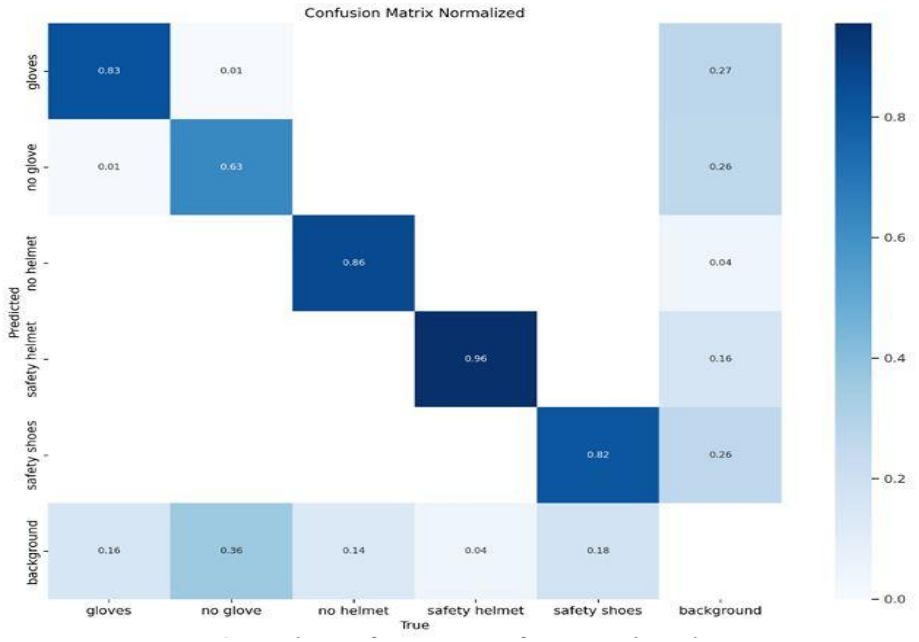

Fig. 9. The confusion matrix for optimal result

5. Conclusions

In conclusion, this study aimed to delve into the capabilities of YOLOv5, 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 YOLOv5 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 YOLOv5 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.

6. 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. (2023): 113051. https://doi.org/10.1016/j.measurement.2023.113051 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

7. 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

8. 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

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. 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

13. 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

14. 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

15. 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

16. 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