كلية علوم الحاسب والمعلومات
قسم تقنية المعلومات

King Saud University
College of Computer and Information Sciences
Department of Information Technology

IT 461 Course Project
Semester-1, 1446H

RISKRADAR

Empowering Safety Through Intelligent Detection.

Student name	ID
Shahd Alfahad	443200491
Shdan Alsheddi	443203081
Shaden Albader	443200524
Hend Al Ghamdi	443200868

Section:72597

Supervised by:

Dr.Yousra Almathami.

Table of Contents

Table of Tables

Table of Figures

Introduction

Construction sites, while essential for infrastructure and development, remain among the most hazardous workplaces, requiring strict safety protocols. Personal Protective Equipment (PPE) - such as helmets, gloves, safety vests, and goggles - serves as the first line of defense, shielding workers from injuries caused by falling objects, heavy machinery, and hazardous substances. The importance of PPE compliance is underscored by statistics from the National Institute for Occupational Safety and Health (NIOSH), which report that wearing helmets alone can reduce head injuries by up to 85% [1]. Despite these measures, however, workplace accidents persist. In recent data from the Occupational Safety and Health Administration (OSHA), approximately 2.8 million workplace injuries are reported each year, with construction sites accounting for a significant portion due to lapses in safety enforcement.

To address this persistent safety issue, RiskRadar focuses on developing an automated solution for monitoring PPE compliance through machine learning. By applying computer vision and deep learning techniques, this project aims to create a model capable of real-time detection of essential PPE items in construction site images. RiskRadar is trained to recognize key safety gear, such as hard hats, gloves, safety vests, and goggles, offering an efficient tool to support continuous safety oversight on large, dynamic worksites.

The primary task is to detect compliance with PPE requirements based on workplace images. Input data includes annotated images from a comprehensive Construction Site Safety Image Dataset, with labels identifying PPE elements such as helmets, masks, and vests. This allows the model to distinguish between compliant and non-compliant individuals. The output is a visual and statistical assessment indicating whether required safety gear is worn.

To ensure the model's reliability and effectiveness, precision, recall, and F1-score metrics are used to assess RiskRadar’s performance. These metrics provide a robust foundation for evaluating the model's accuracy across a wide range of conditions and environments, contributing to the broader vision of safer construction sites through technology.

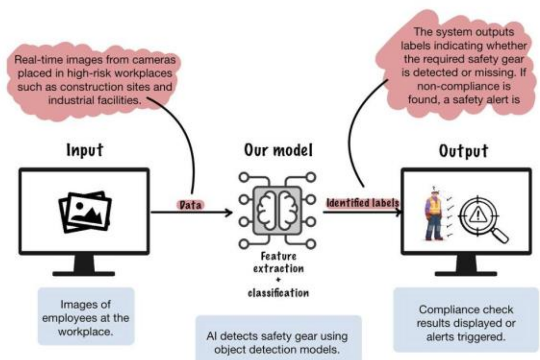

Figure 1: Input and Output of RiskRadar

Related Works

Several studies have focused on improving safety compliance on construction sites through the automated detection of personal protective equipment (PPE). Yange Li et al. [2] proposed a technique to detect safety helmets on workers using convolutional neural networks. Their method leverages the SSD-MobileNet model, which is optimized for real-time object detection while maintaining computational efficiency. By creating a custom dataset of 3,261 images specifically featuring helmet usage, the authors divided the images into training and testing sets to fine-tune the model within the TensorFlow environment. After training, the model exhibited a stable mean average precision (mAP), indicating its reliability in identifying helmets on construction sites. This solution represents a step toward leveraging machine learning for safety management, offering a streamlined and accurate alternative to manual monitoring.

Another relevant study addresses PPE detection in industrial environments such as construction sites and manufacturing plants by developing a comprehensive safety equipment detection model [3]. Using a dataset of 1,641 images, the researchers implemented preprocessing steps such as auto-orientation and resizing to ensure uniform image dimensions of 640x640 pixels. The dataset was divided into training (70%), validation (20%), and test (10%) sets, covering multiple safety gear categories, including helmets and safety vests. Despite class imbalance issues, the model performed effectively in identifying various safety conditions across different PPE classes. Designed for real-time application, the model integrates with existing surveillance systems, enabling continuous monitoring of worker compliance. Additionally, it supports safety training and compliance assessment, which helps organizations meet regulatory standards. However, limitations such as class imbalance and potential difficulties in generalizing to diverse real-world conditions highlight the challenges in adapting these models for broader applications.

A third study explored a deep learning approach for identifying essential safety gear, including helmets, goggles, jackets, gloves, and footwear, using the YOLO v7 object detection algorithm [4]. The authors emphasized the importance of accurate detection of safety gear, noting that a significant portion of workplace incidents stems from inadequate safety equipment usage. Their model was trained on a meticulously annotated dataset and achieved a high mean Average Precision (mAP) of 87.7%, outperforming earlier versions of the YOLO algorithm in detecting safety gear. The study also discussed challenges such as detecting small objects in complex work environments and integrating the model with wearable and augmented reality technology for real-time monitoring. This research contributes to advancing computer vision applications in safety compliance and sets a high-performance benchmark for safety practices in construction, showing the effectiveness of modern deep learning algorithms in improving workplace safety.

These studies collectively demonstrate that computer vision and deep learning techniques, particularly when paired with well-annotated datasets and high-performance models, can significantly enhance PPE detection and safety management in industrial settings. However, they also underscore challenges like class imbalance, generalization to varied environments, and the need for integration with real-time systems, which are essential considerations for improving safety compliance monitoring in dynamic construction and industrial sites.

Dataset

In our RiskRadar project, we rely on a dataset specifically tailored for detecting personal protective equipment (PPE) compliance on construction sites. This dataset, sourced from Roboflow and provided under the CC BY 4.0 License [5], is intended to address the vital need for safety enforcement in construction sites, where the absence of essential protective equipment often results in preventable injuries. By using a dataset with precise, labeled images, RiskRadar can identify key safety gear and determine if workers meet safety standards.

The Construction Site Safety Image Dataset consists of labeled images that accurately capture real-world construction environments, with detailed annotations for various types of PPE and related items. This high level of specificity allows RiskRadar to distinguish between various safety situations, strengthening its capacity as a robust detection model. Each image includes labels that represent both compliant and non-compliant states, providing valuable data for identifying potential safety violations. Key labels include categories such as ‘Hardhat’ and ‘NO-Hardhat,’ which identify whether a worker is wearing a helmet; ‘Mask’ and ‘NO-Mask,’ which indicate the use of respiratory protection; and ‘Safety Vest’ and ‘NO-Safety Vest,’ which show compliance with visibility standards. Additional labels such as ‘Person,’ ‘Gloves,’ ‘Machinery,’ and ‘Vehicle’ capture the presence of individuals, hand protection, heavy machinery, and vehicles within the frame. This thorough labeling approach makes the dataset particularly well-suited for RiskRadar’s objective of real-time safety monitoring by enabling it to detect and differentiate between compliant and non-compliant behavior - an essential feature that is often missing in similar datasets.

The dataset is structured to cover a variety of scenarios and includes training, validation, and test subsets.

Table 1: Overview of the dataset structure

Feature	Description
Number of Images	Training: 2,605; Validation: 114; Test: 82
Number of Classes	10
Total Annotations	Each image includes labels for Hardhat, Mask, NO-Hardhat, NO-Mask, NO-Safety Vest, Gloves, Safety Vest, Person, machinery, vehicle.
Annotation Format	YOLO format (.txt)

These summary statistics offer an overview of the dataset’s structure and balance, providing several examples across different safety gear categories.

We conducted an exploratory data analysis (EDA) to gain insights into our dataset's characteristics, particularly focusing on the distribution of classes across the training, validation, and test sets. Bar charts and pie charts were utilized to visualize these distributions effectively, with the bar charts offering a comparative representation of class frequencies and the pie charts providing a clear proportional view of each subset's composition. These visualizations highlighted the prevalence of classes such as "Person" and "Vehicle," which were consistently represented across all subsets, and provided a deeper understanding of the dataset composition. This analysis offered valuable insights into our dataset structure, supporting an informed and systematic approach to model training and evaluation.

Examples from the dataset illustrate how the model learns to recognize different PPE configurations:

- An image of a worker wearing a helmet, gloves, and safety vest, though lacking a mask.

Figure 2: Example of workers wearing safety gear.

- An image of a workers not wearing a safety gear.

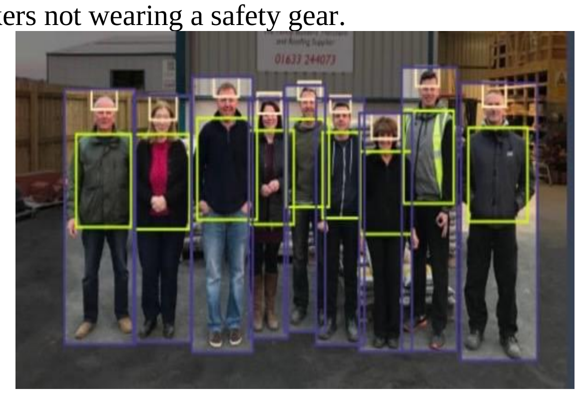

Figure 3: Example of workers not wearing safety gear.

These examples show the variability in PPE usage that RiskRadar will encounter, giving the model a realistic basis for detecting safety compliance.

Methods

In **RiskRadar**, we tackled the challenge of detecting safety equipment in construction site images using two distinct machine learning approaches: **Support Vector Machine (SVM)** for classification and **YOLO11** for object detection. These methods were selected to provide a balance between ease of implementation, clarity, and high performance, catering to the diverse needs of the project.

Support Vector Machine (SVM)

We utilized Support Vector Machines (SVM) as a classifier for distinguishing images based on the presence of specific safety equipment. The choice of SVM was guided by its robustness in high-dimensional spaces and its effectiveness with relatively smaller datasets.

- **Feature Engineering**:
  Images were transformed into feature vectors using Histogram of Oriented Gradients (HOG), a technique that captures essential shape and texture information while reducing computational complexity. This approach is particularly effective for detecting structured objects like safety helmets and vests, where edges and orientations play a key role.
- **Performance Evaluation**:
  The model's performance was assessed using a train-test split strategy, ensuring a reliable evaluation of its generalization ability. Metrics such as accuracy, precision, recall, and F1-score were employed to analyze its ability to classify safety gear effectively.
- **Hyperparameter Tuning**:
  To optimize the model's performance, we experimented with different kernel types (linear, polynomial, and radial basis function). This process allowed us to identify the most suitable kernel for maximizing classification accuracy.

Object Detection using YOLO11

For detecting multiple types of safety equipment simultaneously in construction site images, we employed **YOLO11** (You Only Look Once), a state-of-the-art object detection model. YOLO's ability to perform detection and classification in a single step made it ideal for real-time applications.

- **Baseline Training**:
  The YOLO11 model was initially trained without hyperparameter tuning to establish a baseline for detection performance. Parameters included 10 epochs to prevent overfitting and a batch size of 32 to balance computational efficiency and convergence stability.
- **Performance Evaluation**:
  To evaluate detection performance across multiple safety equipment classes, metrics such as mean Average Precision (mAP), precision, and recall were employed. These metrics provided insights into both the accuracy and reliability of the model's predictions.
- **Hyperparameter Tuning**:
  After establishing the baseline, we applied hyperparameter tuning to enhance the model's detection accuracy. A grid search was conducted to optimize parameters such as:
  - **Learning Rate**: Adjusted to improve convergence rates without destabilizing the training process.
  - **Number of Epochs**: Increased incrementally beyond 10 to identify the optimal balance between learning and overfitting.
  - **Batch Size**: Experimented with larger and smaller batch sizes to assess their impact on training stability and efficiency.

This iterative tuning process resulted in improved detection accuracy, enabling the YOLO11 model to identify safety gear with greater precision and consistency.

The selection of SVM was driven by its interpretability and effectiveness for simpler classification tasks, particularly when feature extraction (e.g., HOG) could be employed to simplify image representations. On the other hand, YOLO11 was chosen for its superior performance in real-time object detection and its ability to handle multiple classes simultaneously, meeting the project's requirement for detecting diverse types of safety equipment within the same image.

Hyperparameter tuning was essential for both models to achieve optimal performance. While SVM tuning focused on kernel types to enhance classification accuracy, YOLO11 tuning targeted learning rates, epochs, and batch sizes to maximize detection precision and efficiency. This dual-method approach ensures a robust and comprehensive solution for both classification and detection challenges.

Experiment

The plan is to initially train the YOLO model without any hyperparameter tuning to establish a performance baseline. This approach allows us to evaluate the model's default capabilities. Following this, we'll conduct a systematic hyperparameter tuning process using **Grid search**, focusing on optimizing key parameters such as learning rate, number of epochs, and batch size. This two-step process ensures we understand the base performance first, then explore improvements through careful parameter adjustments to enhance detection accuracy.

Dataset Splitting and Augmentation:

The dataset, obtained from Roboflow, was pre-split as follows:

- Training Set (93%): 2605 images, designated for model training.
- Validation Set (4%): 114 images, intended for monitoring performance during training and fine-tuning.
- Test Set (3%): 82 images, reserved for unbiased performance evaluation.

Preprocessing:

Roboflow also handled the preprocessing, which included:

- Auto-Orient: To ensure correct image orientation.
- Resize: Images were resized to 640x640 to standardize input dimensions for the YOLO model.
- Class Modifications: The dataset's original classes were reduced, retaining 15 relevant classes while dropping 17 others, optimizing the dataset for the specific task.

Data Augmentation Techniques:

Each training image was transformed into five augmented variants using the following techniques:

- Geometric Augmentations:
    - Horizontal flipping.
    - Zoom: 0%-20%.
    - Rotation: ±12°.
    - Shear: ±2° horizontally and vertically.

- Color Adjustments:
    - Grayscale: Applied to 10% of images.
    - Hue: Adjusted by ±15°.
    - Saturation: Adjusted by ±20%.
    - Brightness: Adjusted by ±25%.
    - Exposure: Adjusted by ±20%.
- Other Techniques:
    - Blur: Up to 0.5 pixels.
    - Cutout: Added six 2%-sized occlusion boxes.
    - Mosaic: Combined multiple images into a single image.

Support Vector Machine (SVM)

The Support Vector Machine (SVM) model served as a baseline for evaluating the dataset. We started by preparing image data for the SVM classifier through feature extraction using the Histogram of Oriented Gradients (HOG) technique. HOG converts each image into a feature vector that summarizes its shape and texture information, making it suitable for classification tasks.

Feature Extraction and Data Preparation

After defining the HOG parameters, we processed the dataset as follows:

- HOG Features: Images were processed using HOG to extract numerical representations that encode edge directions and gradients.
- Data Organization: We iterated through the training, validation, and test sets, loading each image and matching it with its corresponding label to create a labeled dataset with HOG features (X) and class labels (y).
- Dataset Splitting: The dataset was divided into:
    - Training Set (80%): Used for model training.
    - Testing Set (20%): Reserved for evaluation.

Model Training and Evaluation:

We trained the SVM classifier using two different kernels to evaluate its performance. For both kernels, the parameter we use class wight balance applied to handle class imbalances. The default regularization parameter C=1.0 was used.

libraries and the computational resources used during SVM training:

The SVM implementation in RiskRadar leverages key libraries for feature extraction, model training, and performance evaluation. **skimage.feature.hog** is used to extract Histogram of Oriented Gradients (HOG) features as input for the SVM classifier, while **sklearn.model_selection.train_test_split** is utilized to split the dataset into training and test sets. The SVM classifier is trained using **sklearn.svm.SVC**, and the model's performance is evaluated with **sklearn.metrics.accuracy_score** and **classification_report**, providing detailed metrics to assess classification results. These tools enable a comprehensive implementation of SVM-based classification in RiskRadar.

YOLOv11 Without Hyperparameter Tuning

We initially trained the YOLOv11 model without applying any hyperparameter tuning to evaluate its baseline performance on the dataset. This approach provided a foundation to understand how well the model could perform with its default settings and allowed us to identify areas for improvement.

Model Training:
The model was trained on the training set, while validation was performed during training to monitor performance and detect potential overfitting. And for training configuration we use:

- Epochs = 10. providing a basic benchmark without overfitting.
- Batch Size = 32. A standard value that balances memory usage and model update frequency.
- Image Size = 640x640. Images were resized to this standard input size for consistency.

Evaluation metrics for YOLO:
RiskRadar models are evaluated using several key metrics. Precision measures how many of the predicted objects are correct, while Recall assesses how well the model finds all relevant objects in an image. The F1-Score balances Precision and Recall, giving a single measure of a model's accuracy. Other important metrics include Intersection over Union (IoU), which quantifies the overlap between predicted and actual bounding boxes, and mean Average Precision (mAP), which evaluates the model's performance across different IoU thresholds.[6]

YOLOv11 with Hyperparameter Tuning

To optimize the YOLOv11 model's performance, we performed a hyperparameter tuning process aimed at refining key parameters to achieve better detection accuracy and generalization. This tuning was conducted systematically using a grid search approach, where we explored 6 combinations of selected hyperparameters and evaluated their impact on the model's performance metrics.

Hyperparameters Selected for Tuning:

- Batch Size: [32, 64]:
  Batch size 32 was chosen for efficient memory use and faster updates, while batch size 64 was tested for smoother training and better stability, though it uses more memory.

- Epoch: [10,50,100]:

  We chose the epochs range to evaluate the trade-off between training time and model performance. 10 epochs were selected to quickly establish the model's baseline performance and assess its ability to learn patterns in a short training period. 50 epochs provide sufficient time for the model to learn meaningful patterns while balancing the risks of underfitting and overfitting. Finally, 100 epochs allow the model to fully utilize the dataset and capture complex relationships for improved accuracy and generalization. This range ensures comprehensive evaluation and selection of the optimal number of epochs based on performance metrics.

libraries and the computational resources used during YOLO training:

RiskRadar leverages a comprehensive suite of machine learning and computer vision tools, with primary computational support from PyTorch's deep learning framework. Computational resources are dynamically configured to utilize CUDA GPU acceleration when available, falling back to CPU processing if no GPU is detected. Key libraries include Ultralytics YOLO for object detection, NumPy and Pandas for data manipulation, Matplotlib and Seaborn for visualization, Scikit-learn for machine learning utilities, and OpenCV for image processing. The development environment is further enhanced by Roboflow for dataset management and pathlib for file handling. The script is designed to be flexible across computational resources, with built-in capabilities to detect and optimize hardware utilization for efficient model training and inference.

Results and Discussion

In this section, we will showcase and discuss the results of the baseline models, including the **Support Vector Machine (SVM)** and the initial **YOLO11 model** trained with 10 epochs and a batch size of 32. These baseline results provide a benchmark to evaluate the effectiveness of our approach.
We will then compare the performance of these baselines against the **final YOLO model**, which was trained using the optimal parameters identified through grid search (100 epochs, batch size 64). This comparison highlights the impact of hyperparameter tuning and extended training on model performance. Key metrics such as **precision**, **recall**, **mAP@50**, **mAP@50-95**, and **fitness score** will be analyzed in detail to assess improvements in object detection accuracy, generalization, and overall fitness of the final model. The discussion will emphasize how these adjustments influenced the model's ability to detect and classify objects effectively.

Support Vector Machine (SVM)

Model performance:

- Linear Kernel:
  Achieved an accuracy of **41.4%**, performing better on classes with simple, linearly separable boundaries. Showed better generalization compared to the RBF kernel for this dataset.

- RBF Kernel:
  Achieved an accuracy of **40.1%**, designed to capture complex, non-linear relationships in the data but struggled due to the limitations of the extracted features.

The table show the performance for the Best Class (Class 8):

Table 2: Performance of SVM best class

Type	Accuracy	Precision	Recall	F1-score
Linear Kernel	41.4%	45%	88%	59%
RBF Kernal	40.1%	52%	75%	61%

The RBF and linear SVM models demonstrated similar performance, with the linear model achieving an accuracy of 41.4% and the RBF model reaching 40.1%. Both models struggled to fully capture the complexity of the dataset.

To overcome these limitations, we employed YOLOv11, a deep-learning-based model designed to handle complex data.

YOLO11

Model performance:
Here's a brief analytical description of the Yolo11 performance without hyperparameter tuning:

Table 3: Performance of YOLO11n without hyperparameter tuning

Metric	Score	Analysis
Precision	0.84 (84.49%)	The model has a high precision rate, indicating that when it predicts a safety item/person, it's correct 84.49% of the time
Recall	0.61 (60.59%)	The model detects 60.59% of all safety items/persons present.it shows moderate performance in detecting all relevant objects. We could further enhance the model's performance by training for more epochs
mAP50	0.69 (68.75%)	Good overall detection accuracy when using a 50% overlap criterion
mAP50-95	0.36 (36.36%)	This lower score across varying IoU thresholds suggests that while the model performs well at lower thresholds, its performance drops with stricter evaluation criteria, indicating potential issues with detection accuracy.
Fitness Score	0.40 (39.60%):	This overall fitness score reflects a balance of precision, recall, and mAP, suggesting moderate performance.

Final YOLO11 model

Model performance:

After training the final model using the optimal parameters identified through grid search (100 epochs, batch size 64). the model performance metrics demonstrated significant improvement compared to the baseline models results. Below is a table comparing the metrics metrics:

Table 4: Performance of the final YOLO11 model

Metric	Baseline YOLO11 (10 epochs, batch=32)	Final YOLO Model (100 epochs, batch=64)	Improvement
Precision	0.8449 (84.49%)	0.9213 (92.13%)	+7.64%
Recall	0.6059 (60.59%)	0.7412 (74.12%)	+13.53%

mAP@50	0.6875 (68.75%)	0.8165 (81.65%)	+12.90%
mAP@50-95	0.3636 (36.36%)	0.5298 (52.98%)	+16.62%
Fitness Score	0.3960 (39.60%)	0.5585 (55.85%)	+16.25%

Precision: The final model significantly reduces false positives compared to the baseline (+7.64%). This improvement is likely due to the extended training period (100 epochs) and larger batch size (64), allowing the model to learn more robust feature representations.

Recall: The final model detects a larger proportion of true positives, suggesting improved sensitivity (+13.53%). The additional training epochs likely helped the model generalize better, resulting in fewer missed detections.

mAP@50: The improvement in mAP@50 indicates the final model is better at detecting objects with a moderate overlap between predicted and ground truth boxes. This demonstrates a clear improvement in overall detection performance (+12.90%).

mAP@50-95: The sharp increase in mAP@50-95 highlights the final model's ability to handle stricter IoU thresholds, suggesting enhanced bounding box localization accuracy and improved performance on smaller or overlapping objects (+16.62%).

Fitness Score: The fitness score aggregates the overall performance metrics, and its substantial improvement reflects the consistent gains across all aspects of the model's performance (+16.25%).

Training the YOLO11 mode with the best parameters process significantly enhanced the model's ability to accurately detect objects while reducing false positives and improving performance under stricter IoU conditions. The improvements in precision and recall suggest the model is more reliable for practical applications, while the rise in mAP50 and mAP50-95 indicates better performance across different detection thresholds. However, further fine-tuning or advanced techniques, such as additional data augmentation, could push these metrics even higher for greater robustness.

The generalization performance of the models was evaluated by analyzing their behavior on unseen test data. The SVM models, despite utilizing feature-rich representations like HOG, demonstrated limited generalization capabilities. The linear SVM achieved a marginally higher accuracy of 41.4% compared to the RBF kernel's 40.1%, indicating that the extracted features primarily captured linearly separable aspects of the data. However, both models struggled with the dataset's inherent complexity and variations, such as diverse object shapes, occlusions, and lighting conditions. This limitation highlights the challenges of relying solely on hand-crafted features for robust classification in real-world scenarios.

Conversely, the YOLO11 model exhibited a more robust ability to generalize unseen data. Its baseline configuration achieved a mean Average Precision (mAP50) of 68.75%, demonstrating solid detection accuracy under less stringent evaluation criteria. However, the lower mAP50-95 score of 36.36% revealed a drop in performance across stricter IoU thresholds, indicating room for improvement in detecting smaller or partially obscured objects. While the initial results were promising, hyperparameter tuning further improved the model's generalization, showcasing its capability to adapt to the dataset's complexity more effectively than the SVM models. This underscores the advantages of leveraging deep learning for complex tasks requiring high-level feature extraction and detection.

The results reveal a clear advantage of deep learning over traditional methods like SVM for complex tasks. SVM models, limited by the quality of feature extraction, struggled to achieve high accuracy, with both linear and RBF kernels performing similarly. This highlights the challenges of relying on pre-engineered features for non-linear patterns.

YOLO11, however, showed significant improvements, with high precision (84.49%) and better overall performance, especially after extending training to 100 epochs and increasing batch size. While recall and strict localization metrics (mAP@50-95) indicate room for improvement, the model's ability to learn directly from data proved transformative. These findings emphasize the importance of optimizing training configurations and leveraging advanced deep learning techniques for complex, real-world problems.

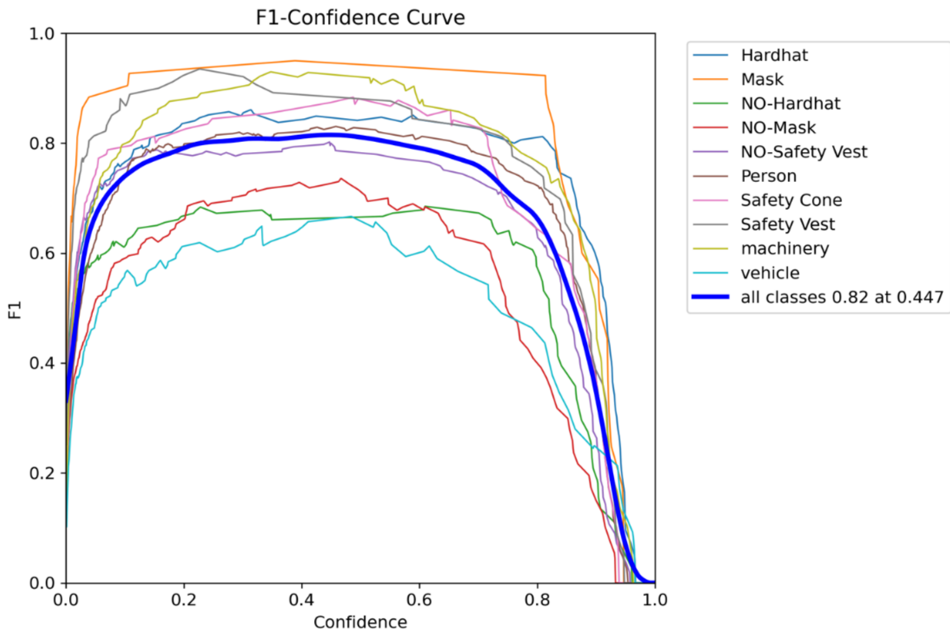

Figure 4: F1-Confidence Curve.

As shown in *Figure 4*, the F1-Confidence Curve illustrates how the F1 score changes with varying confidence thresholds. The model achieves its highest F1 score of 0.82 at a confidence level of 0.447, which indicates the point of optimal performance. The results highlight that the model performs particularly well in classes such as "Machinery" and "Mask," as they maintain consistently high F1 scores. However, the performance for classes like "Vehicle" and "No-Mask" is weaker, as their F1 scores drop sharply at higher confidence levels. This drop suggests that the model struggles to accurately detect these objects, possibly due to limited data representation or more challenging object features.

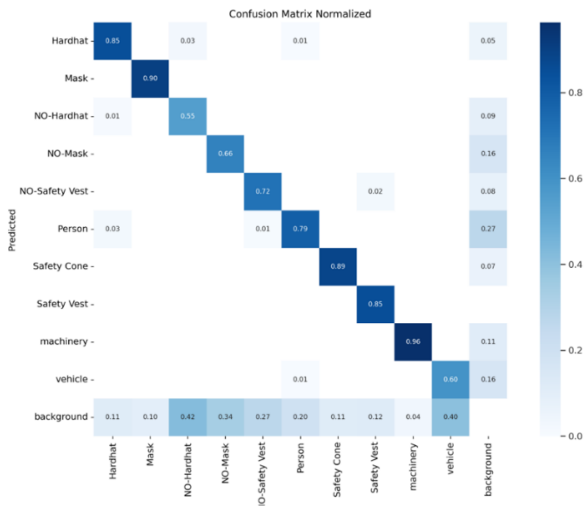

Figure 5:Confusion matrix.

The confusion matrix, displayed in *Figure 5*, provides a breakdown of the model's predictions compared to the actual labels. The matrix shows strong accuracy for classes like "Person," "Machinery," and "Safety Cone," with high counts along the diagonal representing correct predictions. However, some challenges are evident. For example, the class "Vehicle" is frequently misclassified as "Background," highlighting the difficulty in distinguishing between these two. Similarly, the model occasionally confuses "No-Hardhat" and "No-Mask" with their positive counterparts, indicating a need for better differentiation between subtle class variations. These observations suggest that collecting more diverse training data or applying further fine-tuning could help improve performance in these areas.

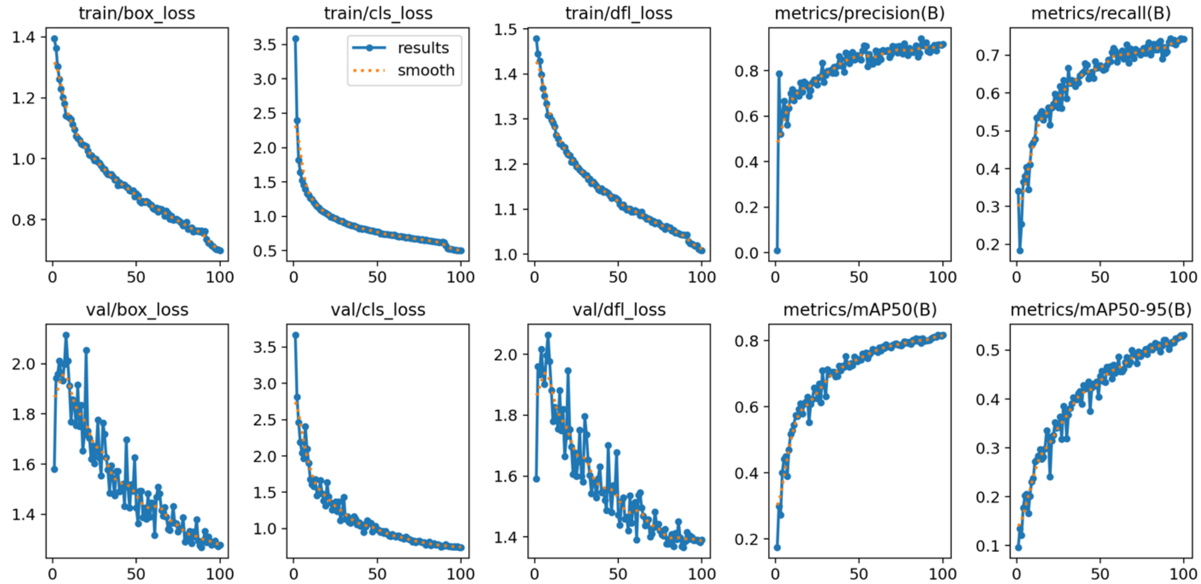

Figure 6:Visualization of training and validation loss.

As shown in *Figure 6*, the training and validation metrics show the impact of fine-tuning on Yolo model's performance, showcasing its enhanced learning and detection accuracy. The top row of the figure highlights a steady reduction in training losses, including box loss, classification loss, and DFL loss, signifying consistent improvement as the training progresses. Similarly, the bottom row reveals a decline in validation losses across epochs, which indicates the model's ability to generalize effectively to unseen data. Furthermore, the precision and recall metrics exhibit a steady upward trend and stabilize at higher values, confirming improved object detection accuracy. Notably, the mAP50 and mAP50-95 scores show considerable growth, with mAP50 nearing 0.9, underscoring the model's strong overall detection performance across varying confidence thresholds.

Conclusion

Our project RiskRadar aimed to enhance safety monitoring on construction sites by leveraging machine learning models to detect personal protective equipment (PPE) in images. We utilized two distinct approaches: Support Vector Machines (SVM) for classification and YOLOv11 for real-time object detection. Our key findings highlighted the strengths and limitations of each method. The SVM models, while providing a baseline for understanding dataset separability, faced significant limitations due to the use of hand-crafted features. The linear kernel achieved a modest accuracy of 41.4%, while the RBF kernel reached 40.1%, demonstrating the challenges of capturing the complexity of the dataset. In contrast, YOLOv11 outperformed SVM, with a baseline mean Average Precision (mAP50) of 68.75% and mAP50-95 of 36.36%. Hyperparameter tuning improved its detection capabilities, showcasing YOLO's strength in real-time object detection tasks.

Despite these successes, the project faced several challenges. One major issue was the limited computational resources, as inconsistent access to high-performance GPUs extended training times and restricted the ability to fully explore model optimization. Additionally, the dataset's variability, including occlusions, small object sizes, and changing lighting conditions, created difficulties for both SVM and YOLOv11. The handcrafted features used in SVM could not capture the fine-grained variations in PPE, making it less suitable for complex detection tasks. These challenges highlighted the need for more powerful resources and a more diverse dataset to improve model performance.

Despite these challenges, the final YOLOv11 model showed substantial improvements over the baseline, particularly in recall and mAP metrics. The model's mAP@50-95 score improved by 16.62%, reflecting its ability to localize objects more accurately. The extended training period, coupled with a larger batch size, helped the model converge more effectively, yielding better generalization. Moving forward, expanding the dataset to include more diverse scenarios, such as varying weather conditions and construction environments, could address challenges like occlusions and environmental changes. Additionally, leveraging advanced techniques like attention mechanisms or transfer learning could further boost the model's performance in detecting smaller or partially occluded safety gear. With more robust computational resources, RiskRadar has the potential to evolve into a more reliable tool for improving safety compliance on construction sites.

Contributions

Student name	Work
Shahd Alfahad	Data – Experiment – Results and discussion
Shdan Alsheddi	Related works – Experiment
Shaden Albader	Introduction – Data – Methods – Conclusion
Hend Al Ghamdi	Experiment - Results and discussion

References

[1] R. Sehsah, A. El-Gilany, and A. M. Ibrahim, “Personal protective equipment (PPE) use and its relation to accidents among construction workers,” La Medicina del lavoro, vol. 111, no. 4, 2020, doi: https://doi.org/10.23749/mdl.v111i4.9398.

[2] Y. Li, H. Wei, Z. Han, J. Huang, and W.-D. Wang, “Deep Learning-Based Safety Helmet Detection in Engineering Management Based on Convolutional Neural Networks,” ResearchGate, Accessed: Sep. 15, 2024. [Online]. Available: (PDF) Deep Learning-Based Safety Helmet Detection in Engineering Management Based on Convolutional Neural Networks (researchgate.net)

[3] Protective Equipment Detector, “Protective Equipment Detector Dataset,” *Roboflow*, 2023. https://universe.roboflow.com/protective-equipment-detector/protective-equipment-detector

[4] S. Islam, S. Shaqib, S. Ramit, S. Khushbu, A. Sattar, and S. Noori, “A Deep Learning Approach to Detect Complete Safety Equipment For Construction Workers Based On YOLOv7,” Jun. 2024, doi: 2406.07707 (arxiv.org)

[5] “Construction Site Safety Object Detection Dataset and Pre-Trained Model by Roboflow Universe Projects,” *Roboflow*. https://universe.roboflow.com/roboflow-universe-projects/construction-site-safety

[6] Henrique Vedoveli, “Metrics Matter: A Deep Dive into Object Detection Evaluation,” *Medium*, Sep. 15, 2023. https://medium.com/@henriquevedoveli/metrics-matter-a-deep-dive-into-object-detection-evaluation-ef01385ec62#:~:text=Precision%2C%20Recall%2C%20and%20F1-Score%20are%20fundamental%20metrics%20used (accessed Sep. 16, 2024).