FEDERAL UNIVERSITY OF TECHNOLOGY, AKURE
JOURNAL OF ENGINEERING AND ENGINEERING TECHNOLOGY

INDIGENOUS WEAPON DETECTION IN IMAGES FOR ENHANCED SECURITY IN NIGERIA USING YOLO APPROACH

[1, 4]Ponnle A. A.*, [2]Udekwe C. N., and [3]Akin-Ponnle A. E.

[1]*Department of Electrical and Electronics Engineering, Federal University of Technology, Akure, Nigeria*
[2]*Department of Information and Communication Technology, Federal University of Technology, Akure, Nigeria*
[3]*Department of Electrical and Electronic Engineering, The Federal Polytechnic, Ile-Oluji, Nigeria*
[4]*Institute of Telecommunications, Aveiro, Portugal*

**Corresponding author: aaponnle@futa.edu.ng*

Ponnle, A. A., Udekwe, C. N., Akin-Ponnle A. E. (2026): Indigenous Weapon Detection in Images for Enhanced Security in Nigeria using YOLO Approach. FUTA Journal of Engineering and Engineering Technology 20(Special), 224-236

Received Date: 15.01.2026 **Accepted Date: 13.03.2026**

Abstract
This work explores the use of the 'You Only Look Once' (YOLO) deep learning model, for real-time weapon detection in images to enhance security. The work addresses the rising threat of insecurity of lives in Nigeria by developing an automated system capable of identifying guns and knives in public spaces, including the local indigenous ones. The primary dataset used consisted of publicly available foreign image datasets of handguns and knives. Also, a small custom dataset was created consisting of images of local indigenous weapons taken within Nigeria. These diverse datasets were pre-processed and used with YOLOv10 model for training, validation and test. Google Colab with GPU acceleration was used for the training. After validation, a mean average precision (mAP) of 64.4% was achieved for all the classes (57% for the knife class and 72% for the handgun class). On the test dataset, an overall precision of 76%, an overall recall of 64%, and an overall F1-score of 70% were achieved; thus, demonstrating some level of success in detecting local indigenous weapons, and detecting handguns better than knives, though challenges such as false positives, false negatives and misclassifications were noted. With larger custom dataset of images of local indigenous weapons for training, better results would be achieved. The work shows that the YOLO-based system can be integrated into existing security infrastructures of the country, with potential to improve existing security measures, and contributing to improved public safety.

Keywords: *detection, guns, images, knives, security, weapon, YOLO.*

Introduction

In recent years, the rise of terrorism, banditry and kidnapping incidents has posed significant threats to public safety and security in Nigeria. These threats often involve the use of weapons which have caused significant harm and panic in various settings, including schools, villages, farms, public transport systems, intercity and interstate transportation, and crowded events. Weapon detection is invaluable in maintaining safety in security-sensitive environments such as airports, schools, religious centres and other public gatherings. Traditional security measures of detecting weapons, including manual physical inspections and use of metal detectors, have limitations in accuracy, speed, scalability, and are error prone (Santos *et al.,* 2024). Security personnel would manually inspect bags and individuals, and video surveillance systems require human operators to monitor footage, which introduces a high risk of human error and missed detections depending on the conditions or state of mind of the human operator (More *et al.,* 2024). These methods, although useful in controlled environments like airports and government buildings, can be time-consuming, require substantial human resources, and have limited ability to detect weapons.

Given the current security landscape of Nigeria and the continual evolution of weapons, the detection of weapons in public and private spaces remains a significant challenge. In the Northwest of Nigeria, banditry, kidnapping and terrorism have caused displacement of many people from their homes and land, hamper free trade, and affected food supplies throughout the country (Akinyede *et al*., 2023). In most advanced countries of the world, information technology (IT) is gaining wide acceptability and usage in the management of crimes (Akinyede *et al*., 2022).

Therefore, in crime scenes where weapons are being used, there is a pressing need in the country for innovative solutions that can enhance the automatic and remote detection of weapons in real time especially in images and video feeds. Though object detectors have made significant advances in accuracy and inference speed, detecting small objects, especially for autonomous weapons in closed circuit television (CCTV) footage, still poses a significant challenge (González *et al.,* 2020). Research into intelligent video surveillance systems has grown with a notable increase in interest in weapon detection and video analytics. With the advancements in the fields of artificial intelligence (AI), interest has increased in developing automated systems for weapon detection. Machine learning (ML) is a field of artificial intelligence that enables systems or machines to learn from data and improve or optimize their performance. With the advent of ML techniques, the field of weapon detection experienced a paradigm shift. ML algorithms like Support Vector Machines (SVMs) with Haar-like features or Histogram of Oriented Gradients (HOG) were used to detect objects in images and video feeds. However, these early ML methods had limitations in handling complex scenarios, such as detecting partially obscured objects or identifying non-standard weapons in diverse environments (Jimenez *et al.,* 2024).

The development of deep learning (DL) has revolutionised the weapon detection landscape. It has shown remarkable success in various object detection tasks. DL is a subset of ML, and it involves training artificial neural networks (ANN) with multiple layers to recognise patterns and make predictions based on large amounts of data. Various models and technologies, including recurrent neural networks (RNNs), long short-term memory (LSTM) networks and convolutional neural networks (CNNs), have been developed and applied to weapon detection in CCTV footages. CNNs and object detection frameworks like Faster Region-based Convolutional Neural Networks (Faster-RCNN) and "You Only Look Once" (YOLO) enable real-time weapon detection with a high degree of accuracy (Santos *et al.,* 2024). These DL-based models allowed for the automatic detection of weapons in images and video streams, significantly reducing the need for human intervention and enhancing response times. CNNs are well-suited for image-related tasks due to their ability to automatically learn spatial hierarchies of features from input images.

Shidik *et al.,* (2019) conducted a comprehensive review of intelligent video surveillance, analysing trends, techniques, and the challenges faced in the field. They noted that there is a gap in the existing research which highlights the need for more comprehensive datasets and enhanced strategies for data processing in surveillance systems. Qi *et al.,* (2021) presented a solution to the problem of gun violence detection by developing a rich dataset specifically designed for video surveillance and a real-time gun detection system. The system combines DL algorithms optimized for edge devices and cloud servers to improve detection accuracy while reducing false positives. Jain *et al.,* (2020) proposed a method for weapon detection using CNN-based models, including single-shot multi-box detectors (SSD) and faster-RCNN architectures. Their research focused on detecting firearms and knives in real-time scenarios, though limitations related to computational speed were identified. Dakalbab *et al.,* (2022) developed a multi-faceted approach using CNN, RNN, and LSTM networks for crime prediction, real-time analysis, and anomaly detection in surveillance videos. Their research contributed to improving public safety by combining predictive analytics with real-time video analysis. Mane (2024) developed a new image dataset for various weapon detection and classification tasks. He used four different models for training and evaluation using different evaluation metrics which are Faster-RCNN with Inceptionv2, Faster-RCNN with Resnet50, SSD with Inceptionv2, and SSD with Resnet50. Experimental analysis showed that the Faster-

RCNN models are superior to SSD models for weapon detection systems.

Among the various DL models developed for object detection in more recent approaches is the YOLO model which stands out for its speed and accuracy (Redmon *et al.,* 2016). The YOLO model divides the image into a grid and predicts bounding boxes and probabilities for each grid cell, thereby enabling localisation of objects within the image. By optimizing YOLO as a single convolutional network, the model processes images at an impressive rate, while maintaining accuracy across a wide range of object categories. The evolution of the YOLO series over the past decade has shown groundbreaking advances in real-time object detection, improving both speed and accuracy through optimized training strategies and architectural improvements (Sapkota *et al.,* 2024). YOLO has been widely adopted for weapon detection due to its ability to process entire images in a single pass, performing object classification and localisation simultaneously, thereby making it faster and more efficient than previous methods (Kumar *et al.,* 2024). YOLOv3 to YOLOv8 have been applied to detect weapons in images, webcams and videos, in dark environments, with email alerts, etc., highlighting the importance of optimising model parameters and preprocessing techniques to enhance detection accuracy (Warsi *et al*., 2019; Reddy *et al*., 2023; Yadav *et al*., 2024; Chunchwar *et al*., 2024; Keerthana *et al*., 2024). Recent advances in YOLO, such as YOLOv9, and YOLOv10, have further enhanced the accuracy and efficiency of weapon detection systems. These models can now detect a wide range of weapons, including firearms, knives, and other dangerous objects, even in cluttered environments or under poor lighting conditions (More *et al.,* 2024). For instance, recent research has proposed integrating YOLO with transformer architectures to improve detection accuracy, especially for small and concealed weapons (Jimenez *et al.,* 2024).

Computer-based weapon detection has witnessed significant advances. However, challenges still remain, particularly in dealing with false positives, false negatives, detecting non-metallic weapons or obscured weapons as well as indigenous weapons, and operating effectively in complex environments like crowded public spaces. These challenges continue to drive research in this area, and further research is ongoing to improve the robustness of these systems while minimising computational costs and false alarms (Santos *et al.,* 2024). Therefore, the aim of this work is to harness the power of YOLO to develop a weapon detection system capable of localizing and identifying guns and knives including the local ones that are indigenous to Nigeria in various environments in images and possibly video feeds, implement and test the system, thus contributing to the prevention of terrorism, kidnapping incidents, and other security issues in Nigeria. The objectives are to collect, annotate, and preprocess a dataset of images of guns and knives in various contexts and conditions including those indigenous to Nigeria; configure and train the YOLO model to detect guns and knives; and evaluate the system.

The YOLO Model

YOLO is a deep learning-based object detection algorithm that can detect more than one object in a single frame. Unlike other detection methods which apply models to image regions multiple times, YOLO frames object detection as a single regression problem rather than a series of classification and localisation tasks (Redmon *et al.,* 2016). This enables real-time detection with high accuracy, thereby making it suitable for tasks that require immediate response. Figure 1 shows the simplified YOLO detection system. efficiency.

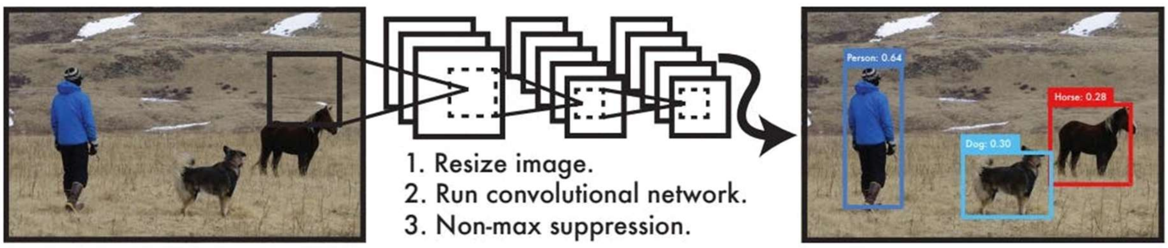


Figure 1: The YOLO Detection System (Redmon *et al.,* 2016).

The key components of YOLO's architecture are:

i. Single Neural Network: YOLO applies a single CNN to the full image, directly predicting bounding boxes with class probabilities in one evaluation.
ii. Grid System: The input image is divided into grids. Each grid cell predicts bounding boxes, confidence scores and class probabilities for the boxes.
iii. Bounding Boxes: Each bounding box includes coordinates, dimensions, and a confidence score that reflects the likelihood of the box containing an object.
iv. Class Prediction: Each grid cell predicts conditional class probabilities, which are multiplied by the confidence scores to yield class-specific confidence scores for each bounding box.

YOLO has evolved through several versions, each bringing improvements in accuracy, speed, and capabilities (Ali and Zhang, 2024). YOLOv1 was released in 2016; YOLOv2 in 2017; YOLOv3 in 2018; and YOLOv4 in 2020 to balance the trade-off between speed and accuracy more effectively (Bochkovskiy *et al*., 2020). In the same year, YOLOv5, developed by Ultralytics was released, though it was not an official continuation by the original authors (Ultralytics, 2020). In 2022, YOLOv6 was released, and in the same year, YOLOv7 was released incorporating advanced training techniques. YOLOv8 was released in 2023; and YOLOv10 in May 2024 with significant advancements over its predecessors to minimise computational overhead while enhancing performance (Wang *et al*., 2024).

Performance Metrics

YOLO models are evaluated using several key performance metrics that assess their detection capabilities. Some relevant ones to this work are discussed briefly:

i. Precision: This is the ratio of true positive (TP) detections to the sum of true positives and false positives (FP). A false positive occurs when the model incorrectly identifies an object or detects something that is not present. High precision means fewer false positives.
ii. Recall: This is the ratio of true positive detections to the sum of true positives and false negatives (FN). High recall means fewer false negatives.
iii. F1 Score: This is a harmonic 'mean' of precision and recall.
iv. Mean Average Precision (mAP): This is the primary metric for object detection and is calculated as the average of average precision (AP) across all classes.
v. Loss Function: This aims to minimise prediction errors during training and it combines three components: classification loss, localisation loss, and confidence loss. The overall loss is a weighted sum of these components.

Materials and Methods

Overview

Building models that generalise well across different environments is a critical challenge in object detection tasks like weapon detection. Figure 2 shows the stages of the processes involved in carrying out this work.

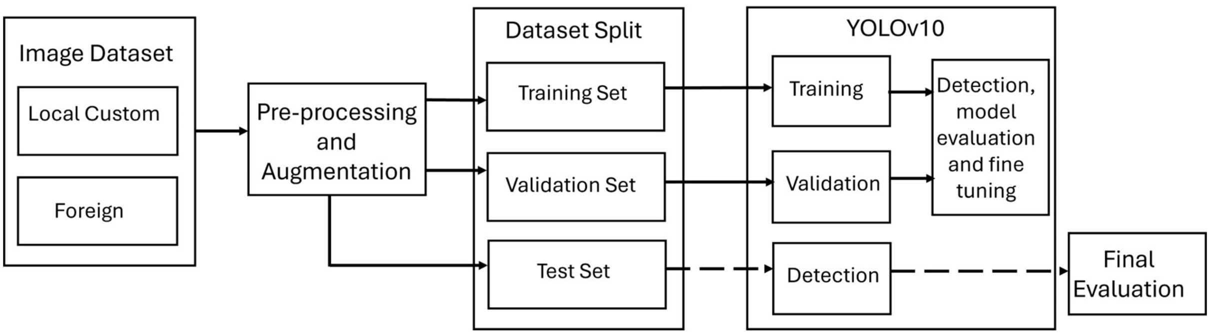
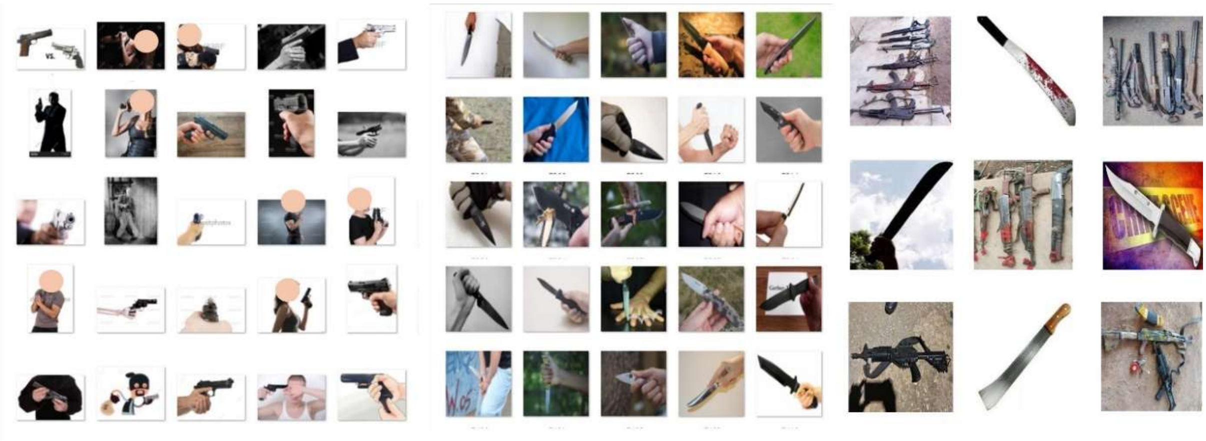


Figure 2: Stages of the processes of this work in block diagram.

The dataset consisted of publicly available image datasets and a small local custom dataset of images of local indigenous weapons in Nigeria. The dataset is of two primary classes: Handguns and Knives. These images were preprocessed, and some image augmentations were applied to them. The dataset was split into training, validation and test sets. The YOLOv10 architecture was chosen due to its balance between speed and precision (https://docs.ultralytics.com/models/yolov10/). The model was trained over several epochs, with hyperparameters being tuned to optimise performance.

Data Collection and Preparation

Data Collection: Creating a robust dataset for weapon detection comes with numerous challenges. These challenges are amplified when dealing with local indigenous weapons of the country**,** which are less standardised and harder to find in existing databases. Most open datasets are focused on internationally recognised weapons, leading to a scarcity of images or videos depicting local indigenous weapons from regions of Nigeria. The dataset used was sourced from publicly available image datasets that included various weapons. The primary sources are images downloaded from Kaggle datasets and Open Images datasets (Kaggle Datasets; Open Image Datasets) from which images for this work were extracted. Also, a small local custom dataset was created consisting of images of local indigenous weapons in Nigeria sourced by the authors from the internet (news outlet sites, social media sites, etc.), camera devices, and from some Nigerian movies in which indigenous local weapons were depicted. In this local custom dataset, attention was paid to the inclusion of some images containing non-weapons, as well as other weapons such as long guns, hunter guns, rifles and machetes. Some images of handguns and knives from the dataset, and some of the local indigenous ones are shown in Figure 3.



(a) (b) (c)

Figure 3: Samples of some of the dataset images showing (a) handguns, (b) knives. (c) Some of the indigenous ones of the local custom dataset created by the authors.

Data Preprocessing: The collected images were preprocessed using Roboflow platform. The following preprocessing operations were performed on the images.

(i) Augmentation: To increase the dataset size and variability, the following image augmentations were applied: (a) Randomly rotating images up to 90 degrees; (b) Applying random scaling transformations to simulate variations in distance; (c) Horizontally flipping images to cover mirrored weapon orientations; and (d) Modifying brightness and contrast to simulate different lighting conditions.

(ii) Annotation: The annotation of images involves labelling the objects within the images. This process was conducted using an annotation tool (CVAT – Computer Vision Annotation Tool) which is an open-source graphical image annotation tool for bounding box annotation. It involved manually drawing bounding boxes around the weapons and assigning them class labels ("Gun" and "Knife"). Each image was annotated with bounding boxes around the weapon objects to mark their locations within the frame, and each box was labeled with the appropriate class. The labelling process ensured the dataset included accurate ground truth annotations for each weapon. The annotations were saved in the YOLO format, which includes a text file for each image with bounding box coordinates and class labels.

Dataset Split: After augmentation, the final dataset used comprised a total of (1,040) images, covering handguns and knives in different poses, backgrounds, lighting conditions, and perspectives to enhance generalisation. Within this dataset is the local custom dataset of 156 images. The final dataset was split into three parts: training, validation and test. The final dataset split is shown graphically in Figure 4.

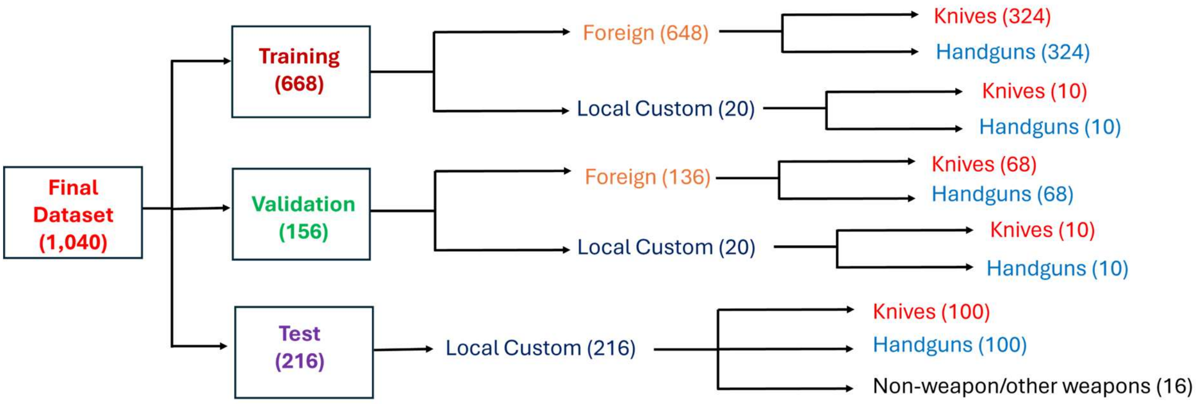


Figure 4: The final dataset

The training set was labelled and used for learning; and the validation set was labelled too and used for tuning hyper-parameters and the model's performance evaluation. The test set was unlabelled and used for final performance evaluation. The training set consists of 668 images of two types of weapons (guns and knives) among which 20 images are from the local custom dataset shared equally (10 of guns and 10 of knives). The validation set consists of 156 images shared equally (78 for guns and 78 for knives). Within the 156 images of the validation set is the local custom dataset of 20 images shared equally. The test set consists of 216 images among which 116 are from the local custom dataset (50 of guns, 50 of knives and 16 of non-weapon/other weapon images). This gives relative proportion of Training: Validation: Test sets to be 65%:15%:20%.

Model Selection and Architecture

The YOLOv10 model was chosen due to its suitability for real-time object detection and its emphasis on speed and accuracy. According to Ultralytics, the model addresses the post-processing and model architecture deficiencies of previous YOLO versions. It eliminates non-maximum suppression (NMS) and optimizes various model components, thereby improving performance and significantly reducing computational overhead (https://docs.ultralytics.com/models/yolov10/; THU-MIG/yolov10). The YOLOv10 model architecture consists of the following components: Backbone, Neck, One-to-Many Head, and One-to-One Head. Its simplified diagram is shown in Figure 5 (Wang *et al*., 2024).

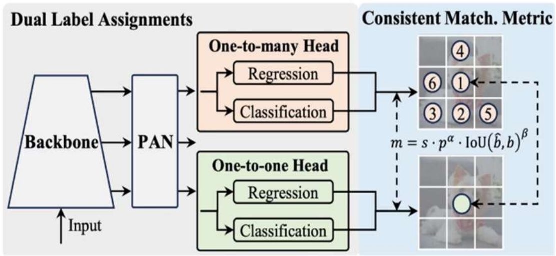


Figure 5: YOLOv10 model components (Wang *et al*., 2024).

Due to resource constraints, the Nano version, YOLOv10-N from the official YOLO GitHub repository was implemented as this version is for extremely resource-constrained environments.

Training, Validation and Test

Training Environment: The model was trained using Google Colab which is an online platform that provides a free GPU environment ideal for deep learning tasks, especially given the computational demands of the YOLOv10 architecture.

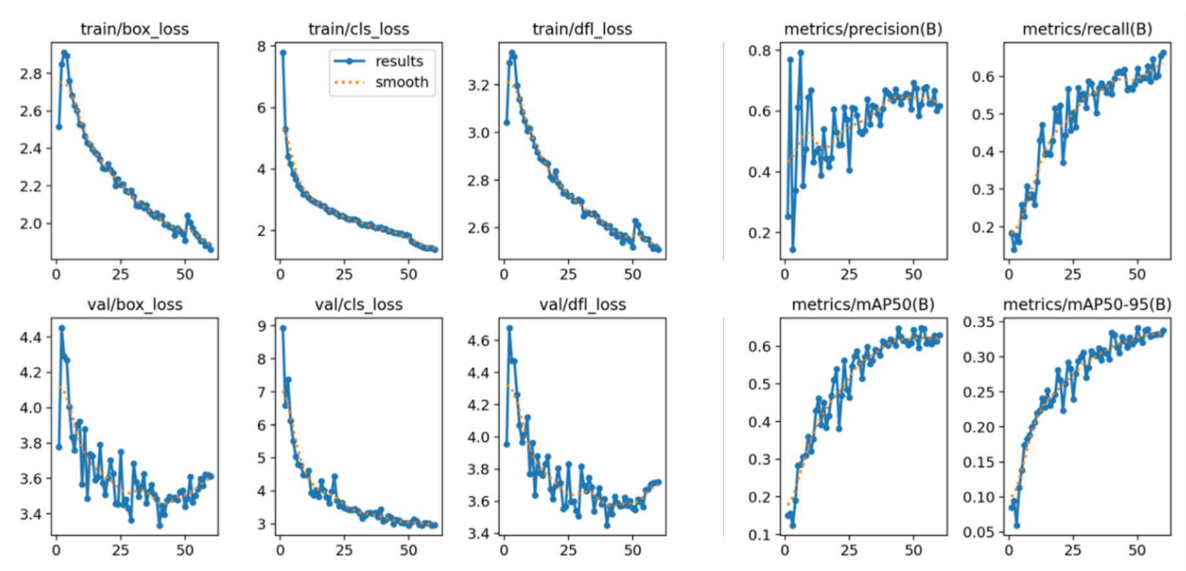
Training Process and Validation: The YOLOv10 model was initialised with pre-trained weights, to allow it to adapt to the new weapon detection task. The learning rate, batch size, and number of epochs were carefully chosen to ensure a balance between convergence, speed and accuracy. The training was carried out with the following parameters: (i) Initial learning rate was set at 0.001 (consequently tuned up through experimentation); (ii) Batch size was set to 16, to balance memory consumption and speed; and (iii) The model was trained for 60 epochs, with early stopping implemented to prevent overfitting. During training and validation, key metrics were monitored to track the model's performance. To optimise and fine-tune the YOLO model, (i) the learning rate was dynamically adjusted based on training progress to improve convergence; (ii) techniques such as rotation, scaling, and flipping were applied to enhance the robustness of the model; and (iii) different hyper-parameter values were experimented to find the optimal configuration. After training and validation, the model's performance was evaluated on the validation set using precision, recall, F1-score and mean Average Precision (mAP) metrics to assess its effectiveness in detecting weapons.

Test Data: After validation, the developed weapon detection model was tested with the unlabelled test dataset and the model's overall performance was evaluated.

Results and Discussion

Model Accuracy and Loss

During training, both accuracy and loss metrics were tracked over each epoch for both the training and validation datasets. Figure 6 illustrates the trends in loss and accuracy for both the training and the validation datasets.



(a) (b)

Figure 6: Plots of model (a) loss and (b) accuracy.

The graphs provide a clear view of the model's learning progression across the epochs and help visualise its convergence towards better detection performance. The training accuracy increased steadily across epochs, reflecting the model's ability to classify and localise weapons in the training set. At the same time, the validation accuracy also improved, indicating that the model was generalising to unseen data.

On the other hand, the training loss decreased consistently with each epoch, showing that the model was progressively refining its predictions on the training data. The validation loss exhibited a similar downward trend but fluctuated slightly, which is common due to the variability in the validation data. Overall, the final training and validation accuracy scores demonstrated that the model was well-optimised.

Model Performance

Figure 7 shows the results of the total training and validation of the model, and Table 1 is the Performance Metric Table for the validation. In the figure and the table, the values for Precision, Recall and the mean Average Precision (mAP) for all classes for the validation correspond to the final mean values attained after the last epoch (60th) in the corresponding plots of Fig. 6(b) for Precision, Recall and mAP50. The mAP for all the classes was 64.4%; the mAP for the Knife class was 57%; and the mAP for the Handgun class was 71.8%. These results show that the model demonstrated some level of success in detecting local indigenous weapons, with better detection of guns than knives. This moderate success is because of very limited datasets used for the entire work both for the training and the validation, and the small number of images of local indigenous weapons.

```
+ Code  + Text
                 all        156        190      0.667      0.602      0.629      0.332

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      59/60      2.98G      1.885      1.426      2.521         15        640: 100% 250/250 [01:49<00:00,  2.29it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 5/5 [00:02<00:00,  2.49it/s]
                   all        156        190      0.601      0.655      0.614      0.333

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      60/60      2.98G      1.861      1.379      2.508         16        640: 100% 250/250 [01:47<00:00,  2.33it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 5/5 [00:03<00:00,  1.42it/s]
                   all        156        190      0.617      0.664      0.629      0.338

60 epochs completed in 1.866 hours.
Optimizer stripped from runs/detect/train/weights/last.pt, 5.8MB
Optimizer stripped from runs/detect/train/weights/best.pt, 5.8MB

Validating runs/detect/train/weights/best.pt...
Ultralytics 8.3.5 🚀 Python-3.10.12 torch-2.4.1+cu121 CUDA:0 (Tesla T4, 15102MiB)
YOLOv10n summary (fused): 285 layers, 2,695,196 parameters, 0 gradients, 8.2 GFLOPs
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 5/5 [00:03<00:00,  1.50it/s]
                   all        156        190      0.687      0.622      0.644      0.341
                 Knife         78         90      0.631      0.611       0.57      0.204
               Handgun         78        100      0.742      0.634      0.718      0.478
Speed: 0.7ms preprocess, 5.0ms inference, 0.0ms loss, 1.4ms postprocess per image
Results saved to runs/detect/train
💡 Learn more at https://docs.ultralytics.com/modes/train
```

Figure 7: Results showing the total training of the model.

Table 1. Performance Metric Table

Class	Images	Instances	Precision	Recall	F1-score	mAP
All	156	190	0.687	0.622	0.655	0.644
Knife	78	90	0.631	0.611	0.621	0.57
Guns	78	100	0.742	0.634	0.688	0.718

With larger datasets for training and validation, and substantial quantities of images of local indigenous weapons used in the training as well, better detection results are expected to be achieved.

Test Data Detection Results

The detection results of the YOLOv10 model on unlabelled test images showed moderate success in identifying guns and knives. In many cases, the model was able to detect them and draw bounding boxes around them with reasonable accuracy. Each detection included a confidence score, reflecting the model's assessment of the likelihood that the detected object was a weapon. Figure 8 shows some detection results for guns and knives, including bounding boxes around detected weapons, along with the confidence scores assigned by the model, and highlighting successful detections, missed detections, and different confidence levels for same images.

(a) (b)

Figure 8: Some images showing results of (a) gun detection, and (b) knife detection.

While the model performed well in relatively straightforward scenarios, such as clear images with distinct weapons, its performance in more complex environments such as images with cluttered backgrounds, poor lighting, or partial occlusions was less consistent. In some instances, the model struggled with detecting weapons taken in black and white (some examples in Figure 8(a)), or types of weapons that are not in the training set (some examples in Figure 8(b)), leading to occasional false negatives or missed detections. Figure 9 shows the Confusion Matrix of the detection results on the test dataset.

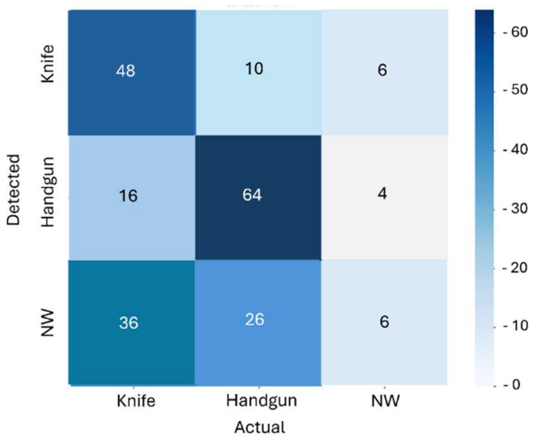


Figure 9: The confusion matrix of detection results of the test dataset showing the distribution of correct, incorrect and missed detections.

In figure 9, NW for the columns (actual) stands for non-weapon images or images with other weapons different from knives and handguns such as machetes, etc., while NW in the rows stands for no detection. The figure shows the number of handgun images that were truly detected, misclassified as knives, and undetected (missed detection). Likewise, the figure shows the number of knife images that were truly detected, misclassified as handguns, and undetected (missed detection). Also, the number of non-weapon/other weapon images that were unclassified are shown (these are not supposed to be detected or classified), and the number of this type of images that were misclassified as handguns and knives are shown.

Number of handgun images, N_h is 100, number of knife images, N_k is 100, and number of non-weapon/other weapon images, N_w is 16. For the handgun class, 64% of the class were truly detected, 10% were misclassified as knife images and 26% were undetected. For the knife class, 48% of the class were truly detected, 16% were misclassified as handgun images, and 36% were undetected. This shows again that the model has better detection of guns than knives. However, the percentage of non-detections for the two classes are considerable. For the non-weapon images, 37.5% were identified as they were, 37.5% misidentified as knife images and 25% misidentified as handgun images.

For the overall analysis, true detection of handgun and knife images is taken as true positive (TP), while non detection of NW images is taken as true negative (TN). Also, non-detection of handgun and knife images is taken as false negative (FN). Detection of NW images as handgun images or knife images is taken as false positive (FP). Also, misclassification of handgun images as knife images and knife images as handgun images are taken as false positive too (FP).

Therefore, from Figure 9, TP, FP, and FN are as shown in Equations (1) – (3).

$$TP = TP_k + TP_h = 48 + 64 = 112 \quad (1)$$

$$FP = FP_k + FP_h = (10 + 6) + (16 + 4) = 36 \quad (2)$$

$$FN = FN_k + FN_h = 36 + 26 = 62 \quad (3)$$

TN = 6

where k and h stand for knife and handgun respectively.

Evaluating Precision, Recall, F1-score and Accuracy as expressed in Equations (4) – (8) respectively for the entire test dataset,

Overall Precision, $$P_{all} = \frac{TP}{TP} \quad (4)$$

Overall Recall, $$R_{all} = \frac{TP}{TP+FN} \quad (5)$$

Overall F1-score, $$F1score_{all} = 2\left(\frac{P_{all} \times R_{all}}{P_{all}+R_{all}}\right) \quad (6)$$

$$= \frac{TP}{TP+\left(\frac{FP+FN}{2}\right)} \quad (7)$$

Overall Accuracy, $$A_{all} = \frac{TP}{TP+FP+FN+T} \quad (8)$$

Substituting values, P_{all} = 0.757; R_{all} = 0.644; F1-score$_{all}$ = 0.696; and A_{all} = 0.546.

Overall, the model was able to detect and distinguish between guns and knives with reasonable precision, although there were some misclassifications and missed detections which are not insignificant. This is responsible for the lower accuracy value. Also, compared with the results in Table 1, these above performance metric values for the test set show a better result than for the validation set, probably owing to larger test set than the validation set. Therefore, this indicates that the model will be effective in many scenarios involving larger datasets; with room for improvement, especially in handling more challenging detection conditions to improve its overall accuracy.

Challenges

Throughout the evaluation of the model for weapon detection, challenges were encountered, particularly with false positives, false negatives, object misclassifications, and different confidence levels for same images. One key issue was the occurrence of false negatives, where actual weapons present in the images were not detected by the model. This often happened in situations where weapons were partially obscured, in motion, appeared in low-contrast conditions (e.g., poor lighting), or image in white and black. These misses are critical, as undetected weapons in real-time surveillance applications could lead to safety risks. Conversely**,** false positives also occurred, where non-weapon objects were mistakenly identified as weapons. In some cases, everyday objects such as tools or household items were incorrectly classified as guns or knives. This was likely due to visual similarities between the shape or texture of certain objects and the weapons in the training data. These false positives could pose problems in real-world scenarios, where incorrect identification might trigger unnecessary alarms.

Additionally, the model occasionally struggled with misclassifications**,** particularly between long guns, cutlasses and knives in the local custom dataset. For example, some objects that should have been identified as guns (long guns) were misclassified as knives, and vice versa. These misclassifications point to the need for more refined training data or further tuning of the model's ability to differentiate between these two classes of weapons.

These challenges highlight areas for further improvement, particularly in enhancing the model's accuracy and reducing its susceptibility to false detections. Future improvements should focus on refining the model to reduce false positives and false negatives, as well as expanding the dataset to include more diverse environments. The dataset used for training and validation primarily focused on guns and knives. Expanding the dataset to include a wider range of weapons and objects in more diverse environments (e.g., crowded public areas, rural settings) could enhance the model's generalisation ability and reduce misclassification errors.

Conclusions

The weapon detection system developed in this work demonstrates the effectiveness of using YOLOv10 for identifying guns and knives, particularly in environments relevant to Nigeria's security context. By leveraging a dataset tailored to include both common foreign and local indigenous weapons, that is, region-specific weapon types that are typically overlooked in standard datasets, the system becomes more robust and applicable to environments such as Nigeria where regional security challenges differ significantly from Western world contexts. On the datasets used, the model achieved good results in terms of precision, recall, and F1-score. Overall accuracy is about 55% due to considerable missed detections of the indigenous weapons, but there is room for further improvement. The system can assist in the prevention of terrorism, kidnappings, and other criminal activities in Nigeria by detecting weapons in real-time.

This work achieved some results, nevertheless, there are some areas where future work would further improve applicability:

(i) For real-time use in public spaces, the model can be optimised for deployment on edge devices or integrated into existing security infrastructure. Further research into reducing the model's computational load without sacrificing accuracy would enable easier integration.

(ii) Future expansion of this work could include the detection of non-lethal weapons (e.g., pepper spray, stun guns, etc.) to address a broader range of security threats.

(iii) The weapon detection system could be integrated into existing CCTV systems or the existing security infrastructure of the country. An interface for security personnel to monitor detected weapons, automated with alarm systems for real-time threat alerts, would make the system more practical for day-to-day security operations. This is essential in high-stress or high-risk environments, allowing security teams to focus on response rather than detection.

Acknowledgement

This work is an advancement on an aspect of a concluded TETFund NRF 2019 project.

References

Akinyede, J., Olebu, C., Ponnle, A., Akinluyi, F., Thompson, A., Dahunsi, O., Alese, B., and Oyinloye, M. (2022). Development of a real-time crime management system in Southwestern Nigeria: The mobile application. *European Journal of Science, Innovation and Technology (EJSIT)*, 2(5), 68 – 82.

Akinyede, J., Olebu, C., Ponnle, A., Akinluyi, F., Thompson, A., Dahunsi, O., Alese, B., and Oyinloye, M. (2023). Development of a software system for realtime management of crime reports in Southwestern Nigeria: The administrative approach. *American Journal of Science, Engineering and Technology (AJSET)*, 8(1), 23-32.

Ali, M. L., and Zhang, Z. (2024). The YOLO framework: A comprehensive review of evolution, applications, and benchmarks in object detection. *Computers*, MDPI, 13(12), 336, 1 - 37.

Bochkovskiy, A., Wang, C. Y., and Liao, H. Y. M. (2020). YOLOv4: Optimal speed and accuracy of object detection. https://doi.org/10.48550/arXiv.2004.10934

Chunchwar, P., Shelare, U., Nagpure, A., Patil, R., Dhole, D., and Shete R. M. (2024). Real-Time weapon detection using YOLOv8 and alert mechanism. *International Journal for Research in Applied Science and Engineering Technology,* 12(4), 2122 – 2129.

Dakalbab, F. M., Abu Talib, M., Waraga, O. A., Nassif, A. B., Abbas, S., and Nasir, Q. (2022). Artificial intelligence and crime prediction: A systematic literature review. *Social Sciences & Humanities Open, Elsevier,* 6(1), 100342s: 1 – 23.

González, J.L.S., Zaccaro, C., Álvarez-García, J.A., Morillo, L.M.S., and Caparrini, F.S. (2020). Real-time gun detection in CCTV: An open problem. *Neural Networks*, 132, 297 – 308.

Jain, H., Vikram, A., Mohana, Kashyap, A., and Jain, A. (2020). Weapon detection using artificial intelligence and deep learning for security applications. *2020 International Conference on Electronics and Sustainable Communication Systems (ICESC)*, Coimbatore, India, 193 – 198. https://doi.org/10.1109/ICESC48915.2020.9155832

Jimenez, J., Toscano, J., Oñate, W., and Caiza G. (2024). Development of firearms and target weapons recognition and alerting system applying artificial intelligence. *WiPiEC Journal - Works in Progress in Embedded Computing Journal*, 10(2), 46 – 50.

Kaggle Datasets. https://www.kaggle.com/datasets

Keerthana, S.M., Sujitha, R., and Yazhini, P. (2024). Weapon detection for security using the Yolo algorithm with email alert notification, *2024 International Conference on Innovations and Challenges in Emerging Technologies (ICICET)*, Nagpur, India, 1-6. https://doi.org/10.1109/ICICET59348.2024.10616365

Kumar, R., Rajpurohit, D. S., Hanief, M., Sharma S., Choudhury T., and Choudhury T. (2024). Detection of criminal activities/criminals through CCTV by live footage analysis. *2024 OPJU International Technology Conference (OTCON) on Smart Computing for Innovation and Advancement in Industry 4.0*, Raigarh, India, 1 – 6. https://doi.org/10.1109/OTCON60325.2024.10687573.

Mane, S.B. (2024). Weapon detection and classification using deep learning. *Journal of Engineering and Technology for Industrial Applications (ITEGAM-JETIA)*, 10(47), 19 – 26.

More, P., Patil, S., and Pattanshetti, T. (2024). Real-time violence and weapon detection and alert system. *Research Square*. Information on https://doi.org/10.21203/rs.3.rs-4389485/v1.

Open Images Dataset and Extensions. https://storage.googleapis.com/openimages/web/

Qi, D., Tan, W., Liu, Z., Yao, Q., and Liu, J. (2021). A dataset and system for real-time gun detection in surveillance video using deep learning, *In proceedings of 2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Melbourne, Australia, 667 – 672. https://doi.org/10.1109/SMC52423.2021.9659207.

Reddy, D., Poojitha, M., Puspalalitha, G., Reddy, M.V.V., Kumar, K.A., and Sree, G.N. (2023). Weapon identification using YOLOv5 algorithm, *International Journal of Science and Research (IJSR)*, 12(11), 550 – 564.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You Only Look Once: Unified, real-time object detection, *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, USA, 779 – 788. https://doi.org/10.1109/CVPR.2016.91.

Santos, T., Oliveira, H., and Cunha, A. (2024). Systematic review on weapon detection in surveillance footage through deep learning. *Computer Science Review*, *Elsevier*, 51(C), 100612.

Sapkota, R., Flores-Calero, M., Qureshi, R., Badgujar, C., Nepal, U., Poulose, A., Zeno, P., Vaddevolu, U. B. P., Khan, S., Shoman, M., Yan, H., and Karkee, M. (2025). YOLO advances to its genesis: a decadal and comprehensive review of the You Only Look Once (YOLO) series. *Artificial Intelligence Review*, 58, Article 274. https://doi.org/10.1007/s10462-025-11253-3

Shidik, G. F., Noersasongko, E., Nugraha, A., Andono, P. N., Jumanto, J., and Kusuma, E. J. (2019). A systematic review of intelligent video surveillance: Trends, techniques, frameworks, and datasets. *IEEE Access*, 7, 170457-170473.

THU-MIG/yolov10. Information on https://github.com/THU-MIG/yolov10 (accessed 15 January 2025).

Ultralytics, YOLOv5 GitHub, (2020). Information on https://github.com/ultralytics/yolov5.

Wang, A., Chen, H., Liu, L., Chen, K., Lin, Z., Han, J., and Ding, G. (2024). YOLOv10: Real-time end-to-end object detection. *38th Conference on Neural Information Processing Systems (NeurIPS 2024)*. ArXiv, abs/2405.14458. 1-25.

Warsi, A., Abdullah, M., Husen, M.N., Yahya, M., Khan, S., and Jawaid, N. (2019). Gun detection system using Yolov3, *2019 IEEE International Conference on Smart Instrumentation, Measurement and Application (ICSIMA)*, Kuala Lumpur, Malaysia, 27-29 August, 1-4.

Yadav, P., Gupta, N., and Sharma, P.K. (2024), Robust weapon detection in dark environments using Yolov7-DarkVision, *Digital Signal Processing, Elsevier*, 145, 104342, 1 - 11.

YOLOv10: Real-time end-to-end object detection. Information on https://docs.ultralytics.com/models/yolov10.