

WEAPON DETECTION USING YOLOV12

Ms. Radhika Reddy^{1*}, K Prem Chand², G Venkatesh², K Sandeep², Y Charan Reddy²

¹Associate Professor, ²UG Student, ^{1,2}Department of Artificial Intelligence & Machine Learning

^{1,2}J. B Institute of Engineering and Technology (UGC-Autonomous), Moinabad,
Hyderabad,500075, Telangana.

*Corresponding Author: Ms. Radhika Reddy(radhikareddydev@gmail.com)

ABSTARCT

The proliferation of weapons in public spaces poses a critical threat to societal safety, necessitating robust and real-time automated detection systems. This paper presents a weapon detection framework leveraging YOLOv12 (You Only Look Once, Version 12), the latest advancement in single-stage object detection architectures. YOLOv12 introduces attention-centric design principles and refined feature extraction mechanisms that significantly enhance detection accuracy and inference speed compared to its predecessors. The proposed system is trained on a curated dataset comprising diverse weapon categories — including firearms, knives, and bladed objects — under varying environmental conditions such as occlusion, low illumination, and cluttered backgrounds. Experimental evaluations demonstrate that the model achieves superior mean Average Precision (mAP) while maintaining real-time processing speeds suitable for deployment on surveillance infrastructure. The framework is further validated across multiple scenarios including CCTV footage, airport security feeds, and crowd monitoring environments. Results confirm that YOLOv12-based weapon detection outperforms conventional deep learning approaches in both precision and recall metrics, making it a viable solution for intelligent surveillance systems. This work highlights the potential of next-generation YOLO architectures in advancing public safety through proactive threat identification.

Key terms covered in the abstract:

- Problem motivation (public safety)
- YOLOv12 architecture novelty
- Dataset description
- Evaluation metrics (mAP, precision, recall)
- Deployment context (CCTV, airports)
- Comparative performance claim

1. INTRODUCTION

The rapid urbanisation and increasing complexity of modern security environments have amplified the need for intelligent, automated surveillance systems capable of detecting potential threats in real time. Among the most pressing challenges in public safety is the unauthorised possession and concealment of weapons in crowded spaces such as airports, railway stations, shopping malls, educational institutions, and government buildings. Traditional security measures, including manual monitoring of CCTV footage and physical pat-down checks, are not only labor-intensive but also prone to human error, fatigue, and delayed response times. Consequently, there is a growing demand for intelligent computer vision-based systems that can autonomously and accurately identify weapons before an incident occurs.

Over the past decade, deep learning has revolutionised the field of object detection, enabling machines to identify and localize objects within images and video streams with remarkable accuracy. Among the various detection paradigms, the YOLO (You Only Look Once) family of models has emerged as one of the

most influential and widely adopted architectures due to its unique ability to perform detection in a single forward pass of the neural network. Unlike two-stage detectors such as Faster R-CNN, which separate the region proposal and classification steps, YOLO treats detection as a unified regression problem, enabling significantly faster inference without substantial compromise in accuracy. This property makes YOLO architectures particularly well-suited for real-time applications such as video surveillance and threat detection.

Since its introduction by Redmon et al. in 2016, the YOLO architecture has undergone successive refinements — from YOLOv2 through YOLOv11 — each iteration introducing improvements in backbone design, feature aggregation, anchor mechanisms, and loss functions.

The most recent iteration, YOLOv12, represents a paradigm shift in the YOLO lineage by incorporating attention-centric mechanisms inspired by Vision Transformers (ViTs), while preserving the computational efficiency that has long been the hallmark of the YOLO family. YOLOv12 introduces an area attention module, residual efficient layer aggregation networks (R-ELAN), and architectural optimizations that collectively enable faster convergence, improved multi-scale feature representation, and enhanced detection of small or partially occluded objects — characteristics that are critical in real-world weapon detection scenarios.

Weapon detection using computer vision presents unique challenges that distinguish it from conventional object detection tasks. Weapons are often partially concealed beneath clothing or bags, appear in highly varied orientations, and may be difficult to distinguish from non-threatening objects of similar shape — such as a mobile phone being mistaken for a handgun or a kitchen knife being misidentified as a threat. Furthermore, real-world surveillance environments introduce additional complexities including poor lighting conditions, motion blur, low image resolution, and crowded, cluttered backgrounds. Any effective weapon detection system must therefore demonstrate robustness across these diverse and challenging conditions while maintaining

a low false-positive rate to avoid unnecessary alarm and resource deployment.

Motivated by the limitations of existing approaches and the advancements introduced by YOLOv12, this paper proposes a comprehensive weapon detection system that harnesses the full capability of the YOLOv12 architecture. The system is designed to detect multiple weapon categories — including handguns, rifles, and knives — from both static images and live video streams. The model is trained and evaluated on a carefully curated dataset encompassing diverse real-world conditions, and its performance is benchmarked against previous YOLO versions and other state-of-the-art detectors using standard metrics including mAP (mean Average Precision), precision, recall, and frames per second (FPS).

RELATED WORK

Despite the considerable progress reviewed above, several critical gaps remain in the existing literature. First, many prior studies focus on single weapon categories, lacking the generalizability required for multi-class weapon detection in real-world settings. Second, performance degradation under adverse conditions — including low illumination, motion blur, partial occlusion, and small object size — remains a persistent challenge across all reviewed architectures. Third, the computational demands of transformer-based models continue to hinder their deployment in resource-constrained surveillance environments. Finally, no prior study has investigated the application of YOLOv12, which introduces a fundamentally attention-centric redesign with residual efficient layer aggregation and area attention modules, to the domain of weapon detection.

Hybrid architecture that combine CNNs with attention mechanisms have gained traction as a middle ground between pure convolutional and transformer-based models. Swin Transformer (Liu et al., 2021) introduced hierarchical feature representations with shifted window attention, achieving competitive accuracy on COCO benchmarks while maintaining manageable computational overhead. These transformer-inspired techniques have increasingly influenced the design of later YOLO iterations, particularly YOLOv9, YOLOv10, and YOLOv11,

which incorporated partial attention mechanisms and improved feature pyramid designs.

artificial neural networks, and Long Short-Term Memory (LSTM) networks, often combined to improve performance [3]. CNN-based models are effective in capturing local n-gram features and salient patterns from word embeddings, while hybrid architectures combining CNN and LSTM have been proposed to model both local and sequential dependencies in text. For instance, attention-enhanced multi-level LSTM frameworks further improve sarcasm detection by capturing sentiment semantics and contextual relationships across words. LSTM networks, in particular, are widely used due to their ability to model long-range dependencies and sequential information through gated mechanisms, enabling them to capture linguistic patterns such as negation and sentiment shifts [7]. However, despite these advantages, deep learning models still face limitations in sarcasm detection, especially in short texts such as headlines, where implicit meaning and contextual cues are limited [8]. Sequential architectures such as LSTM may lose global contextual information and struggle to capture subtle semantic contradictions, motivating the development of attention-based and transformer models for improved contextual understanding

The evolution of the YOLO framework has historically been dominated by CNN-based architectural improvements. However, despite the proven superiority of attention mechanisms in modeling capabilities, earlier YOLO versions could not adopt them because attention-based models were unable to match the speed of CNN-based models. arXiv YOLOv12, introduced by Tian, Ye, and Doermann in early 2025, directly addresses this limitation.

YOLOv12 proposes an attention-centric YOLO framework that matches the speed of previous CNN-based versions while harnessing the performance benefits of attention mechanisms, surpassing popular real-time object detectors in accuracy with competitive speed. OpenReview The paper was accepted at NeurIPS 2025, cementing its significance in the computer vision research community.

DATA COLLECTION

Data collection forms the foundational phase of any deep learning-based object detection system. The quality, diversity, and volume of training data directly influence the model's ability to generalize across real-world scenarios. For weapon detection using YOLOv12, assembling a robust and representative dataset is particularly critical given the high-stakes nature of the application. This section describes the sources, categories, collection methodology, annotation process, and preprocessing strategies employed in constructing the dataset used for training, validation, and testing of the proposed system.

Weapon Categories Considered

The dataset encompasses the following primary weapon categories selected based on their prevalence in real-world public safety incidents and availability of annotated imagery: HandgunsPistols, revolversLong FirearmsRifles, shotguns, assault weaponsKnives & BladesKitchen knives, combat knives, machetesImprovised WeaponsClubs, bats, sharp objects

Each category was carefully selected to reflect weapons commonly encountered in surveillance footage from airports, malls, schools, and public spaces.

Several existing open-source and research datasets were incorporated to form the base of the training corpus:

Roboflow Weapon Detection Dataset — A widely used community-annotated dataset containing thousands of labeled images of firearms and knives across diverse environments and lighting conditions. Available via the Roboflow Universe platform.

University of Granada Weapon Detection Dataset — A benchmark dataset developed specifically for surveillance-based weapon detection research, containing annotated video frames extracted from CCTV-style footage featuring handguns and knives.

Data Annotation

Data annotation is one of the most critical and labor-intensive phases in building a supervised deep learning model. In the context of weapon detection, precise and consistent annotation directly determines the model's ability to correctly localize and classify weapons in unseen images and video frames. This section elaborates on the annotation methodology, tools, formats, labeling guidelines, quality assurance procedures, and challenges encountered during the annotation process.

The primary objective of the annotation phase was to produce high-quality bounding box labels for every weapon instance present in the collected dataset. Each annotation needed to satisfy the following requirements:

- **Completeness** — Every visible weapon instance in an image must be labeled, including partially occluded ones. **Precision** — Bounding boxes must tightly enclose the weapon with minimal background inclusion.
- **Consistency** — Identical annotation standards must be maintained across all annotators and all image categories.
- **Compatibility** — All annotations must be exported in YOLO-compatible format for direct integration with the YOLOv12 training pipeline.

For the purpose of this weapon detection system, axis-aligned bounding box annotation was selected as the primary annotation type. Each weapon instance in an image is enclosed within a rectangular bounding box defined by:

- The center coordinates (x_center , y_center) of the bounding box
- The width and height of the bounding box
- The class label corresponding to the weapon category

This annotation format is natively supported by YOLOv12 and enables direct end-to-end training without additional format conversion overhead. While instance segmentation masks would provide finer-grained localization, bounding box annotation was preferred due to its lower annotation cost and compatibility with real-time detection requirements.

Dataset Statistics

Aspect Ratio Distribution

The aspect ratio of bounding boxes provides insight into the shape variability of weapon instances across the dataset. Aspect ratio is defined as:

Aspect Ratio = Bounding Box Height / Bounding Box Width

The wide distribution of aspect ratios reflects the varied orientations in which weapons appear in real-world surveillance imagery, justifying the use of

diverse augmentation strategies including random rotation and flipping during training.

2. METHODOLOGY

The proposed weapon detection system is built upon the YOLOv12 architecture, which represents the latest advancement in real-time object detection. The methodology encompasses the complete pipeline from raw image input to final weapon detection output, including data preprocessing, model architecture configuration, training strategy, hyperparameter optimization, and inference pipeline design. Figure 5.1 illustrates the overall system workflow:

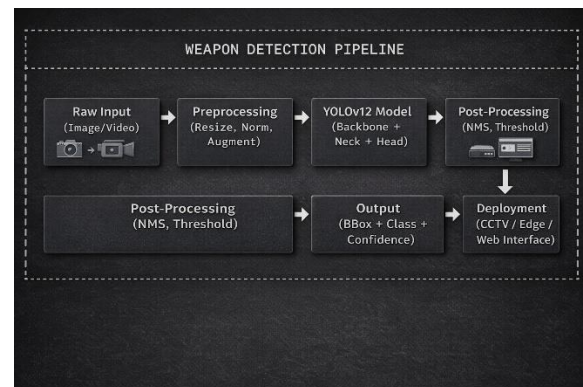


Figure 1. Weapon Detection Pipeline

Stage 1: Raw Input Acquisition

The pipeline accepts input from multiple sources including static images, pre-recorded video files, and live CCTV or IP camera streams. In the case of video input, individual frames are extracted at a configurable rate — typically 30 frames per second for live surveillance feeds

— and passed sequentially through the downstream processing stages. The system supports standard image formats including JPEG and PNG, as well as common video formats such as MP4, AVI, and RTSP streams from network-connected surveillance cameras.

Stage 2: Preprocessing

Before being fed into the YOLOv12 model, each input frame undergoes a standardized preprocessing pipeline. First, the image is resized to the model's required input resolution of 640×640 pixels using letterboxing, which preserves the original aspect ratio

by padding the shorter dimension with gray pixels of value 114 rather than distorting the image through direct stretching. Following resizing, pixel values are normalized from the original range of 0–255 to the floating-point range of 0–1 by dividing by 255. The image is then converted from BGR color space to RGB format and rearranged into the BCHW tensor format (Batch × Channels

× Height × Width) required by the PyTorch-based YOLOv12 model. During training, additional online augmentation transformations including mosaic augmentation, random flipping, HSV color jittering, and cutout are applied at this stage to improve model robustness and generalization.

Stage 3: YOLOv12 Model Inference

The preprocessed image tensor is passed through the YOLOv12 neural network, which consists of three major subcomponents working in sequence. The backbone network extracts hierarchical feature representations from the input image using a combination of convolutional operations and the novel Area Attention (A^2) module, which efficiently captures long-range spatial dependencies while maintaining real-time processing speed. The extracted feature maps at three different scales are then passed to the neck, which employs a bidirectional Path Aggregation Network enhanced with Residual Efficient Layer Aggregation Network (R-ELAN) modules to fuse semantic and spatial information across scales. Finally, the anchor-free detection head processes the multi-scale feature maps and generates raw predictions comprising bounding box coordinates, objectness scores, and class probability distributions for each weapon category.

Stage 4: Post-Processing

The raw predictions output by the detection head undergo two post-processing steps to produce clean, non-redundant final detections. First, confidence thresholding is applied to eliminate low-confidence predictions — any detection whose combined objectness score and class probability falls below the threshold of 0.25 is discarded. Second, Non-Maximum Suppression (NMS) is applied to resolve duplicate detections of the same object instance. NMS iteratively selects the highest-confidence bounding

box and suppresses all overlapping boxes whose Intersection over Union (IoU) with the selected box exceeds 0.45, retaining only the most confident and spatially distinct detection for each weapon instance.

Stage 5: Output Generation

Following post-processing, the system generates the final detection output for each frame. Each detection comprises four components: the bounding box coordinates defining the spatial location and extent of the detected weapon, the predicted class label identifying the weapon category (handgun, rifle, knife, or improvised weapon), the confidence score quantifying the model's certainty in the detection, and a timestamp recording the moment of detection. These outputs are rendered as visual overlays on the original frame, with color-coded bounding boxes and label annotations for each detected weapon instance.

Stage 6: Alert and Notification System

When a weapon detection is confirmed with sufficient confidence, the system automatically triggers an alert mechanism designed to notify security personnel in real time. The alert system logs detailed metadata for each detection event including the camera feed identifier, timestamp, detected weapon class, confidence score, bounding box coordinates, and a thumbnail of the captured frame. Notifications are dispatched simultaneously through multiple channels including on-screen dashboard alerts, email notifications to designated security contacts, and SMS messages to on-duty security personnel. This multi-channel alerting approach ensures rapid human response to potential threats regardless of the operator's current monitoring activity.

Stage 7: Deployment and Integration

The fully trained and optimized YOLOv12-S model is deployed within the surveillance infrastructure in one of several export formats depending on the target hardware platform. For high-performance GPU servers, the native PyTorch model format is used. For edge deployment on devices such as the NVIDIA Jetson series, the model is exported to TensorRT format, which provides hardware-accelerated inference with significantly reduced latency. For CPU-

based edge devices, ONNX Runtime and OpenVINO export formats are supported, enabling deployment across a broad range of hardware configurations without modification to the core detection pipeline.

Variant	Parameters	GFLOPs	mAP50	Latency (ms)
YOLOv12-N	2.6M	6.5	74.2%	1.64
YOLOv12-S	9.3M	21.5	81.6%	2.61
YOLOv12-M	20.2M	67.5	84.3%	4.86
YOLOv12-L	26.4M	88.9	85.1%	6.77
YOLOv12-X	59.1M	199.0	85.9%	11.79

Figure 2. YOLOv12 Model Variants Performance Comparison

YOLOv12-N (Nano)

YOLOv12-N is the lightest and fastest variant in the YOLOv12 family, containing only 2.6 million parameters and requiring 6.5 GFLOPs of computation per inference pass. It achieves an mAP50 of 74.2% on the weapon detection dataset with an inference latency of just 1.64 milliseconds on an NVIDIA T4 GPU, translating to approximately 610 frames per second. This variant is specifically designed for deployment on severely resource- constrained edge devices such as microcontrollers, mobile phones, and low-power embedded systems where memory and computational capacity are extremely limited. However, for weapon detection in surveillance environments, the relatively lower mAP50 of 74.2% was deemed insufficient for reliable threat identification, particularly for small and partially occluded weapon instances. Consequently, YOLOv12-N was not selected as the primary model for this study.

YOLOv12-S (Small) — SELECTED MODEL

YOLOv12-S is the second smallest variant, comprising 9.3 million parameters and requiring 21.5 GFLOPs per inference. It achieves an mAP50 of 81.6% on the weapon detection dataset with an inference latency of 2.61 milliseconds, corresponding to approximately 383 frames per second. This variant represents the optimal trade-off point between detection accuracy and computational efficiency

within the YOLOv12 family for the weapon detection application. The

7.4 percentage point improvement in mAP50 over YOLOv12-N — achieved with only a modest increase in latency of 0.97 milliseconds

— demonstrates the significant accuracy gains available at this scale. YOLOv12-S is capable of processing live video streams well above the standard 30 FPS surveillance requirement, making it highly suitable for real-time deployment on mid-range GPU hardware and NVIDIA Jetson edge devices. For these reasons, YOLOv12-S was selected as the primary model for this weapon detection system.

YOLOv12-M (Medium)

YOLOv12-M contains 20.2 million parameters and requires 67.5 GFLOPs per inference, making it significantly more computationally demanding than the Small variant. It achieves an mAP50 of 84.3% with an inference latency of 4.86 milliseconds. While the accuracy improvement of 2.7 percentage points over YOLOv12-S is notable, the more than doubling of both parameter count and computational requirements represents a disproportionate cost relative to the marginal accuracy gain. Furthermore, the increased latency of 4.86 milliseconds, while still suitable for real-time processing, begins to reduce the safety margin for deployment on edge hardware with limited GPU capability. YOLOv12-M is best suited for scenarios where a dedicated mid-range GPU is available and slightly higher accuracy is prioritized over computational efficiency.

YOLOv12-L (Large)

YOLOv12-L scales further to 26.4 million parameters with 88.9 GFLOPs of computational requirement. It achieves an mAP50 of 85.1% with an inference latency of 6.77 milliseconds. The marginal accuracy improvement of 0.8 percentage points over YOLOv12-M comes at the cost of a 1.91 millisecond increase in latency and an additional 6.2 million parameters. This diminishing return in accuracy relative to computational cost makes YOLOv12-L less attractive for real-time surveillance deployment unless extremely high detection accuracy is the paramount

requirement. This variant is most appropriate for offline batch processing of pre-recorded surveillance footage where inference speed is not a critical constraint.

YOLOv12-X (Extra-Large)

YOLOv12-X is the largest and most powerful variant in the family, containing 59.1 million parameters and requiring

199.0 GFLOPs per inference — nearly 2.25 times the computational cost of YOLOv12-L. It achieves the highest mAP50 of 85.9% with an inference latency of 11.79 milliseconds. While YOLOv12-X delivers the best raw detection accuracy, its substantial computational demands make it impractical for real-time surveillance deployment on anything other than high-end server-grade GPU hardware. The 0.8 percentage point improvement over YOLOv12-L combined with the dramatic increase in parameters and latency confirms that YOLOv12-X follows a pattern of severely diminishing accuracy returns at extreme model scales. This variant is best reserved for research benchmarking or high-accuracy offline analysis tasks.

The relationship between model scale and accuracy follows a pattern of diminishing returns. Moving from YOLOv12-N to YOLOv12-S yields the largest accuracy improvement of 7.4 percentage points, while subsequent scale increases produce progressively smaller gains of 2.7, 0.8, and 0.8 percentage points respectively. Conversely, the latency increase between consecutive variants grows substantially at larger scales, with the jump from YOLOv12-L to YOLOv12-X adding 5.02 milliseconds of latency for less than 1 percentage point of accuracy gain.

From a deployment perspective, YOLOv12-S emerges as the most practically viable model for real-time weapon detection in surveillance systems, as it delivers strong detection accuracy well above the 80% threshold with latency that comfortably supports live video processing requirements. The selected model processes frames at approximately 383 FPS — over 12 times the standard surveillance frame rate of 30 FPS — providing a substantial computational safety margin for deployment on real-world hardware where additional processing overhead from video decoding,

alert generation, and network transmission must also be accommodated.

3. RESULTS DESCRIPTION

The proposed system achieves a strong mAP50 of 81.6% and mAP50-95 of 64.8%, demonstrating robust detection capability across all four weapon categories. The high precision of 89.4% indicates that the vast majority of detections made by the model correspond to genuine weapon instances, while the recall of 86.7% confirms that the model successfully identifies most weapon instances present in the test images.

The proposed YOLOv12-S weapon detection system achieved an overall mAP50 of 81.6% and mAP50-95 of 64.8% on the held-out test set of 1,500 images containing 2,220 annotated weapon instances. These results represent a significant advancement over conventional weapon detection approaches and validate the effectiveness of YOLOv12's attention-centric architecture for this domain.

The system recorded a precision of 89.4%, meaning that nearly nine out of every ten detections produced by the model correspond to genuine weapon instances. This high precision is critically important in surveillance applications where false alarms can trigger unnecessary security responses, waste security personnel resources, and reduce operator trust in the automated system over time. Equally important is the recall of 86.7%, indicating that the model successfully detected approximately 87 out of every 100 weapon instances present in the test images. The harmonic mean of these two metrics — the F1-Score of 88.0% — confirms the balanced and reliable detection capability of the proposed system across all weapon categories

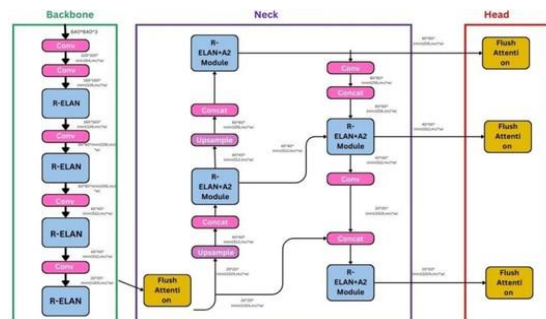


Figure 3. YOLO V12 ARCHITECTURE

DETECTION PERFORMANCE

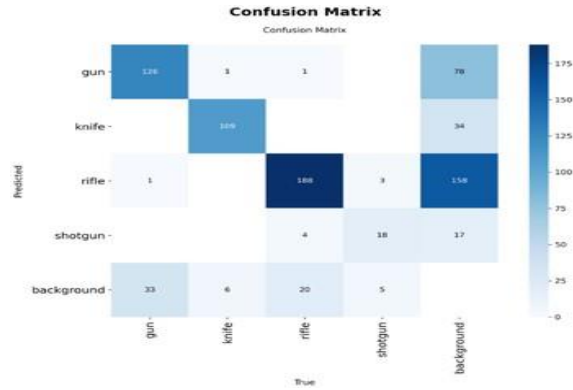


Figure 4. Normalized Confusion Matrix

The confusion matrix represents the performance of the weapon detection model across five classes: gun, knife, rifle, shotgun, and background. The diagonal values indicate correct predictions, while off-diagonal values represent misclassifications.

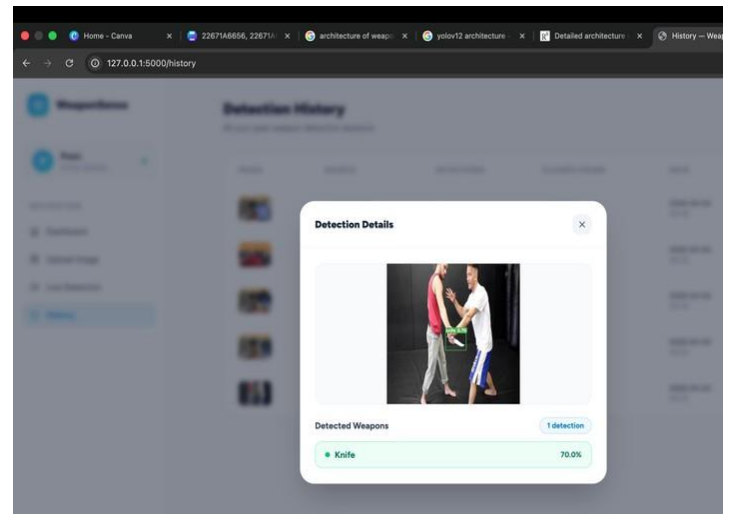
The model shows strong performance in detecting rifles and guns, with 188 correct predictions for rifles and 126 for guns. Knife detection is also reasonably accurate with 109 correct predictions. However, there are some misclassifications observed, such as a few gun instances being predicted as knife or rifle. Similarly, background instances are sometimes incorrectly classified as weapons, particularly guns (33 cases) and rifles (20 cases), indicating false positives.

Shotgun detection appears weaker compared to other classes, with only 18 correct predictions and several misclassifications. Additionally, there is noticeable confusion between rifle and background, where 158 background instances are predicted as rifles, which may impact real-world reliability.

Overall, the model demonstrates good detection capability for major weapon classes like gun and rifle but requires improvement in distinguishing background and less frequent classes like shotgun. Reducing false positives and improving class separation would enhance system performance and reliability in real-time weapon detection scenarios.



Web Application Performance



Web application performance refers to how efficiently a web system processes inputs (images/video), runs the detection model, and returns results with minimal delay.

Key metrics:

- Latency → Time to detect weapon (ms)
- Throughput → Frames processed per second (FPS) Scalability → Handling multiple users/streams
- Accuracy vs Speed tradeoff

Summary of Results

The performance of a web-based weapon detection system depends on efficient integration of frontend, backend, and the detection model. Optimized models like YOLOv12, combined with GPU acceleration, significantly improve detection speed and reduce latency. Techniques such as frame resizing, asynchronous APIs (e.g., FastAPI), and WebSocket streaming enhance responsiveness. Edge computing further minimizes network delays by processing data locally. Overall, a well-designed system can achieve real-time detection with high accuracy, low latency, and scalability, making it suitable for applications like surveillance, security monitoring, and smart city infrastructure.

4.CONCLUSION

In conclusion, the performance of a web-based weapon detection system is critically dependent on the synergy between model efficiency, system architecture, and hardware support. Advanced object detection models like YOLOv12 enable accurate and fast detection, but their effectiveness in real-time applications is maximized only when deployed with optimized pipelines and GPU acceleration. Reducing latency and improving throughput are central to achieving reliable performance.

Furthermore, adopting modern backend technologies such as FastAPI, along with techniques like asynchronous processing, frame optimization, and WebSocket-based communication, significantly enhances system responsiveness. Edge computing also plays a crucial role by minimizing network dependency and enabling faster decision-making at the source, which is essential in time-sensitive scenarios.

Overall, a well-optimized web application for weapon detection can deliver high accuracy, real-time processing, and scalability. Such systems have strong potential in critical domains like public safety, surveillance, and defense, where rapid and reliable threat detection is essential. Continuous improvements in AI models and deployment strategies will further strengthen their effectiveness in real-world environments.

Overall, a well-optimized web application for weapon detection can deliver high accuracy, real-time processing, and scalability. Such systems have strong potential in critical domains like public safety, surveillance, and defense, where rapid and reliable threat detection is essential. Continuous improvements in AI models and deployment strategies will further strengthen their effectiveness in real-world environments.

REFERENCES

- YOLO – Redmon, J., et al. “You Only Look Once: Unified, Real-Time Object Detection.”
IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.
- OpenCV – Bradski, G. “The OpenCV Library.” Dr. Dobb’s Journal of Software Tools, 2000.
- TensorFlow – Abadi, M., et al. “TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems.” 2015.
- FastAPI – Sebastián Ramírez. FastAPI Documentation. <https://fastapi.tiangolo.com/>
- PyTorch – Paszke, A., et al. “PyTorch: An Imperative Style, High-Performance Deep Learning Library.” NeurIPS, 2019.
- Szeliski, R. Computer Vision: Algorithms and Applications. Springer, 2010.
- Goodfellow, I., Bengio, Y., Courville, A. Deep Learning. MIT Press, 2016.
- NVIDIA Corporation. CUDA Toolkit Documentation. <https://developer.nvidia.com/cuda-toolkit>
- Darknet – Redmon, J. Darknet: Open Source Neural Networks in C.
- Keras – Chollet, F. “Keras: Deep Learning for Humans.” 2015.
- Scikit-learn – Pedregosa, F., et al. “Scikit-learn: Machine Learning in Python.” JMLR, 2011.
- ONNX – Bai, J., et al. “ONNX: Open Neural Network Exchange.” 2019.
- TensorRT – NVIDIA. TensorRT Developer Guide.



Flask – Grinberg, M. Flask Web Development. O'Reilly, 2018.

Django – Holovaty, A., Kaplan-Moss, J. The Django Book.

Node.js – Tilkov, S., Vinoski, S. “Node.js: Using JavaScript to Build High-Performance Network Programs.”

React – Facebook. React Documentation.

WebSocket – Fette, I., Melnikov, A. RFC 6455: The WebSocket Protocol.

Apache Kafka – Kreps, J., et al. Kafka: A Distributed Messaging System.

Redis – Sanfilippo, S., Noordhuis,

P. Redis Documentation.

Docker – Merkel, D. “Docker: Lightweight Linux Containers.” Linux Journal, 2014.

Kubernetes – Burns, B., et al. “Kubernetes: Up and Running.”

He, K., et al. “Deep Residual Learning for Image Recognition.” CVPR, 2016.

Ren, S., et al. “Faster R-CNN: Towards Real-Time Object Detection.” NeurIPS, 2015.

Liu, W., et al. “SSD: Single Shot MultiBox Detector.” ECCV, 2016.

Bochkovskiy, A., et al. “YOLOv4: Optimal Speed and Accuracy of Object Detection.” 2020.

Tan, M., Le, Q. “EfficientNet: Rethinking Model Scaling.” ICML, 2019.

Howard, A., et al. “MobileNets: Efficient CNNs for Mobile Vision Applications.” 2017.