AUTOMATIC THREAT DETECTION IN X-RAY IMAGES USING DEEP LEARNING

HALİL UĞUR BAYEZİT

FEBRUARY 2025

ÇANKAYA UNIVERSITY

GRADUATE SCHOOL
ELECTRICAL-ELECTRONICS ENGINEERING DEPARTMENT
ELECTRICAL-ELECTRONICS ENGINEERING MASTER'S THESIS

AUTOMATIC THREAT DETECTION IN X-RAY IMAGES USING DEEP LEARNING

HALİL UĞUR BAYEZİT

FEBRUARY 2025

ABSTRACT

AUTOMATIC THREAT DETECTION IN X-RAY IMAGES USING DEEP LEARNING

BAYEZİT, HALİL UĞUR
ELECTRICAL-ELECTRONICS ENGINEERING MASTER'S THESIS

Supervisor: Prof. Dr. MEHMET REŞİT TOLUN
February 2025, 93 Pages

Automated object detection in X-ray baggage screening is critical for maintaining security and operational efficiency in high-throughput environments such as airports. Traditional methods often struggle with the unique challenges of X-ray imagery, including overlapping objects and low contrast. Recent developments in deep learning, and the YOLO (You Only Look Once) architecture specifically, have been especially promising for real-time object detection.

This thesis benchmarks the performance of the newest YOLO variants—YOLOv8, YOLOv9, and YOLOv10—on three X-ray baggage datasets that are widely used—CLCXray, PIDXray, and SIXray. The comparison is done based on important parameters such as detection accuracy, inference speed, and computational cost to evaluate their applicability in real-time applications. Comprehensive experiments are implemented to evaluate their performance in object detection in dense and complicated scenes with an emphasis on the trade-off between detection accuracy and processing time.

Results indicate that YOLOv10 performs best overall with higher accuracy and quicker inference at low computational complexity. YOLOv8 and YOLOv9 also show competitive performance with benefits under certain circumstances. Results indicate the efficacy of recent YOLO models in meeting the requirements of real-world X-ray

baggage screening systems and their readiness for deployment in operational security settings.

This research provides a detailed analysis of cutting-edge object detection models, contributing immensely to the knowledge of their application in real-world scenarios and paving the way for the creation of more sophisticated automated security systems.

Keywords: Automated Object Detection, X-Ray Baggage Screening, Deep Learning Techniques, Real-Time Detection

`

ÖZET

AUTOMATIC THREAT DETECTION IN X-RAY IMAGES USING DEEP LEARNING

BAYEZİT, HALİL UĞUR
ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ YÜKSEK LİSANS TEZİ

Danışman: Prof. Dr. MEHMET REŞİT TOLUN
Şubat 2025, 93 Sayfa

X-ray bagaj taramalarında otomatik nesne tespiti, güvenliği ve yüksek hacimli ortamlar (örneğin, havaalanları) gibi yerlerde operasyonel verimliliği sağlamak için kritik bir öneme sahiptir. Geleneksel yöntemler, X-ray görüntülemenin benzersiz zorlukları olan üst üste binen nesneler ve düşük kontrast gibi durumlarla baş etmekte genellikle yetersiz kalmaktadır. Derin öğrenmede son yıllarda yaşanan gelişmeler, özellikle YOLO (You Only Look Once) çerçevesi, gerçek zamanlı nesne tespiti için dikkate değer bir potansiyel göstermiştir.

Bu tez, en güncel YOLO modellerinden YOLOv8, YOLOv9 ve YOLOv10'un en yaygın kullanılan üç X-ray bagaj veri kümesi üzerindeki performansını incelemekte ve karşılaştırmaktadır. Bu veri kümeleri şunlardır; CLCXray, PIDXray ve SIXray.

Araştırma, bu modelleri algılama doğruluğu, çıkarım hızı ve hesaplama verimliliği gibi temel metrikler açısından değerlendirmekte ve gerçek dünya uygulamalarına uygunluklarını belirlemeyi amaçlamaktadır. Çalışma, karmaşık ve dağınık ortamlardaki nesneleri tespit etme yeteneklerini kapsamlı deneylerle incelemekte, algılama hassasiyeti ve işlem hızını dengeleme üzerinde yoğunlaşmaktadır.

Sonuçlar, YOLOv10'un genel olarak en iyi performansı sergilediğini, üstün doğruluk ve daha hızlı çıkarım süreleri sunarken düşük hesaplama karmaşıklığını koruduğunu ortaya koymaktadır. YOLOv8 ve YOLOv9 da belirli senaryolarda öne

`

çıkan güçlü yönleriyle rekabetçi bir performans sergilemektedir. Bulgular, en yeni YOLO modellerinin gerçek dünya X-ray bagaj tarama sistemlerinin gereksinimlerini karşılamada oldukça etkili olduğunu ve operasyonel güvenlik ortamlarında kullanım potansiyeli taşıdığını gösteriyor.

Bu çalışma, nesne algılama modellerinin kapsamlı bir incelemesini sunuyor, pratik uygulamalar hakkında değerli bilgiler sağlıyor ve otomatik güvenlik sistemlerindeki gelişmelerin temelini oluşturuyor.

Anahtar Kelimeler: Otomatik Nesne Tespit Teknikleri, X-Ray Bagaj Tarama, Derin Öğrenme Teknikleri, Gerçek Zamanlı Algılama

`

ACKNOWLEDGEMENT

I would like to express my deepest gratitude to my beloved father Mustafa Bayezit, who inspired me, played the biggest role in my choice of profession and guided me, and to my dear mother Gülsüm Bayezit, who has always been by my side and supported me unconditionally.

I would like to express my deepest gratitude to my dear sister Görkem Bayezit Arlı and my first teacher Dr. Ahmet Çağrı Arlı for paving my way in my professional life, showing me the right path and guiding me.

I would like to express my eternal gratitude to my beloved wife/fiancée Elif Solmaz, who has been with me even in my most difficult moments, giving me faith and motivation when I wanted to give up.

I would like to express my respect and thanks to Prof. Dr. Mehmet Reşit Tolun for his support and valuable contributions to the completion of this study.

`

TABLE OF CONTENTS

`

`

LIST OF FIGURES

`

`

LIST OF SYMBOLS AND ABBREVIATIONS

ABBREVIATIONS

CNN	: Convolutional Neural Network
FN	: False Negative
FP	: False Positive
TN	: True Negative
TP	: True Positive
RCNN	: Region-based Convolutional Neural Network
SSD	: Single Shot MultiBox Detector
YOLO	: You Look Only Once
SIXray	: A Large-Scale Security Inspection X-ray Benchmark
PIDray	: A Large-scale X-ray Benchmark for Real-World Prohibited Item Detection
CLCXray	: Cutters and Liquid Containers X-Ray Dataset

`

CHAPTER I
INTRODUCTION

1.1 BACKGROUND

In modern security systems, particularly airport baggage scanning, the requirements of effective and efficient object detection have increased significantly. Major airports scan millions of bags on a daily basis; Heathrow Airport alone scans over 200,000 bags on peak travel days [1]. With the increasing number of travelers and the growing security threats, automated systems are the need of the hour to identify threats accurately and within a reasonable time frame. Delay or inaccuracy in screening is extremely expensive and has serious consequences, including disruption in passenger flow, economic loss to airlines, and most important, possible compromise of security through the passage of dangerous or prohibited materials.

X-ray imaging serves to play a significant part in this regard by facilitating the identification of hidden objects within bags. Yet, some unique challenges of X-ray images, including overlapping objects, low contrast, and cluttered noisy backgrounds, may adversely affect detection performance [2] Conventional image processing methods and visual inspections are not adequate to manage increasing levels and complexities of baggage screening. Therefore, sophisticated deep learning techniques have been embraced to augment the object detection speed and accuracy [3].

1.2 ADVANCES IN OBJECT DETECTION

Object detection techniques, including the R-CNN family [4], SSD [5], and YOLO [6], have greatly improved the state of the art for computer vision. The R-CNN family of algorithms, including Fast R-CNN and Faster R-CNN, pioneered the approach of utilizing region proposals to zoom in on areas of the image where there are probable objects, then regressing and classifying the bounding box. These models are accurate but are computationally costly and too slow for real-time uses.

Likewise, SSD integrates the advantages of region proposal and dense prediction at various feature scales and provides a quicker solution than R-CNN.

`

Nevertheless, SSD also cannot attain real-time velocities required in dynamic high-throughput applications such as baggage screening at the expense of precision.

In comparison, the YOLO (You Only Look Once) model is a tremendous improvement that predicts bounding boxes and class probabilities with a single forward pass, which is extremely fast and adequate for real-time systems. YOLO's speed vs. accuracy trade-off is an important benefit for low latency systems as well as high detection accuracy systems.

1.3 WHY FOCUS ON YOLO

Given the requirements of baggage screening in terms of high accuracy, scalability, and processing in real-time, this work is interested in object detection models of the YOLO family only. Compared to SSD-based and R-CNN-based methods, YOLO is specifically designed to be used in applications in real-time, providing a unique optimization of detection capability in terms of computational efficiency. The one-time nature of YOLO means that it processes photographs in milliseconds, making it of great interest in applications that require strict performance, such as high-throughput security screening.

The design of YOLO has evolved tremendously over time. From the first YOLO model [6], it evolved to become YOLOv2 [7], YOLOv3 [8], YOLOv4 [9], to newer models of YOLOv5, YOLOv6, and YOLOv7, each of which was better in terms of accuracy, speed, and generalizability. The newer models, such as YOLOv8, YOLOv9, and YOLOv10, continued to set detection limits without compromising their defining real-time efficiency.

The choice of YOLOv8, YOLOv9, and YOLOv10 in this work is guided by their high processing power, precision, and architectural efficacy over their counterparts. The new versions apply advanced mechanisms such as more efficient anchor-free mechanisms, work distribution flexibility, and more efficient fusion of features that highly boost detection precision, to a great extent in hard cases such as riddled baggage in imaging in X-ray imaging and low-contrast imaging. The versions also endeavor to lower requirements in terms of processing to be highly resourceful to be applied in applications in real-time in devices that often get utilized in deployment in security systems in place. Compared to their counterparts, YOLOv8, YOLOv9, and YOLOv10 provide the best precision-inference time ratio to provide accurate and swift

`

object detection that is paramount in ensuring high throughput in airport security scenarios.

For this reason, this work looks at YOLO models of object detection in X-ray baggage imagery in isolation, for they better capture the requirements of working in real-time that security systems absolutely need.

1.4 PROBLEM STATEMENT

The detection of objects in baggage in X-ray imagery is hindered by The presence of overlapping and occluded objects. The large variance in object size, object shape, and object orientation. Low imagery resolution and noisy imaging conditions.

Existing object detection models, SSD and R-CNN, even though highly efficient in usual computer vision scenarios, lack high enough real-time capabilities to be employed in dynamic scenarios such as baggage scanning. The YOLO suite of models is a more promising candidate in that it is faster without compromising detection accuracy.

1.5 OBJECTIVES

The objectives of this research are; to evaluate the performance of state-of-the-art YOLO models (YOLOv8, YOLOv9, YOLOv10) in detecting objects within X-ray images. To compare these models against specialized X-ray detection frameworks such as SIXray, CLCXray, and PIXray. To analyze the trade-offs between detection accuracy, speed (frames per second), and generalization across diverse baggage scenarios. To identify the most suitable object detection framework for real-time baggage screening.

1.6 RESEARCH QUESTIONS

How do YOLO models (v8, v9, v10) perform in terms of accuracy and speed compared to specialized X-ray detection models? What are the key trade-offs between general-purpose object detection frameworks and domain-specific X-ray detection models? Can YOLO models outperform or complement specialized X-ray models in challenging baggage screening scenarios?

`

1.7 CONTRIBUTIONS

This study contributes to the field of object detection and baggage screening; Conducting a comprehensive performance comparison between YOLO models and specialized X-ray detection frameworks. Benchmarking these models on real-world X-ray datasets to evaluate their accuracy, speed, and generalization capabilities. Proposing insights into the applicability of general-purpose models like YOLO for domain-specific tasks in security systems. Providing a roadmap for optimizing detection frameworks for real-time baggage screening applications.

1.8 THESIS STRUCTURE

This thesis is structured as follows:

- Chapter 2: Literature Review — A detailed exploration of object detection techniques, including YOLO and specialized X-ray models, along with their applications in baggage screening.
- Chapter 3: Methodology — An explanation of the dataset preparation, model selection, training pipeline, and evaluation metrics used in this study.
- Chapter 4: Experimental Results — Presentation of quantitative and qualitative results, including performance comparisons and ablation studies.
- Chapter 5: Discussion, Conclusion— The chapter discusses key findings, limitations, and real-world implications, summarizes contributions and research outcomes, and provides recommendations for future advancements in methodology and practical applications.

`

CHAPTER II
LITERATURE REVIEW

2.1 OVERVIEW OF OBJECT DETECTION TECHNIQUES

2.1.1 Traditional Methods

It's all about how object detection algorithms worked in the past, through hand-crafted features and machine learning. These techniques are the most important insights into the historical growth of object detection before deep learning approaches were actively used.

2.1.1.1 Feature Extraction-Based Methods

Early object detection methods, which extracted new features using applicable methods for describing objects contained within images. These approaches are fundamental to understanding the transition toward deep learning techniques. Among the most commonly used feature extraction methods were Scale-Invariant Feature Transform (SIFT) [10], Speeded-Up Robust Features (SURF) [11], and Histogram of Oriented Gradients (HOG) [12]. To enable the object classification and detection, these functionalities are paired with various machine learning techniques like Support Vector Machines (SVM).

2.1.1.1.1 HOG + SVM

One of the most widely used classical object detection techniques employed a combination of HOG descriptors and SVM classifiers. HOG descriptors extracted image gradient information and were extremely useful for shape detection, for instance, in pedestrian detection. The HOG features were classified into their appearance using SVMs. This was a technique proposed by Dalal and Triggs [12], was the standard for object detection problems before the invention of deep learning techniques.

`

2.1.1.1.2 SIFT and SURF

Both SIFT [10] and SURF [11] are both feature extraction techniques that detect distinctive key points and descriptors invariant to scale, rotation, and partial occlusion. These techniques were quite successful at detecting objects in images with different orientations, scales, and visibility. Both SIFT and SURF were seminal in many early object detection and matching applications.

2.1.1.1.3 Traditional Machine Learning: SVM for Object Detection

Support Vector Machines (SVM) were commonly used in combination with feature extraction methods to classify objects in an image. The application of the SVM classifier to the features extracted from the object enabled the binary classification of the objects. These traditional methods provided solid results for simpler object detection tasks but were computationally expensive and struggled with real-time applications.

2.1.2 Deep Learning-Based Methods

With the rise of deep learning, traditional methods have been largely replaced by models that learn hierarchical features from data.

2.1.2.1 Convolutional Neural Network (CNN)

Convolutional Neural Networks (CNNs) are designed to recognize patterns directly from images by mimicking the human visual system. CNNs have a strong impact on tasks like image classification, object detection, and segmentation. [13]

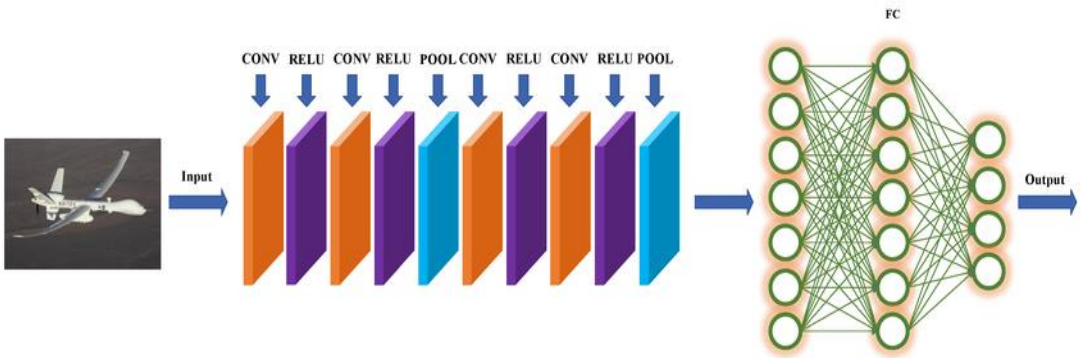


Figure 1: CNN Architecture

`

2.1.2.1.1 CNN Workflow

2.1.2.1.1.1 Input Image

The CNN processes an input image, which can be of arbitrary size. The image is typically resized to a fixed dimension for compatibility with the network.

Example: A 224x224 pixel RGB image. [13]

2.1.2.1.1.2 Convolutional Layer

The convolutional layer applies filters (also known as kernels) to the input image or the feature maps from previous layers. Filters detect basic features such as edges, textures, and corners.

Convolution involves sliding the filter across the input and performing element-wise multiplication with the local region of the image. The matematical insights :

$$(I * K)(x, y) = \sum_{m} \sum_{n} I(m, n) * K(x - m, y - n) \quad (2.1)$$

Where I is the input image, K is the filter and (x,y) are the coordinates of the output feature map. A feature map where each pixel represents a detected feature. [13]

2.1.2.1.1.3 Activation Function (ReLU)

After the convolution, an activation function (typically ReLU) is applied to introduce non-linearity to the network, provide the model's ability to recognize intricate patterns.

The matematical insights:

$$ReLU(x) = \max(0, x) \quad (2.2)$$

The model accommodates non-linear relationships in the feature using ReLU. [13]

2.1.2.1.1.4 Pooling Layer (Max Pooling)

Pooling layers reduce spatial dimension of the feature maps to reduce computational load and avoid overfitting.

`

A common approach of max pooling, where the pooling window scans the feature map that extracting the maximum value from each section. The matematical insights:

For a region R of size p x p :

$$P(R) = \max(R) \tag{2.3}$$

where R is the region of the feature map under the pooling window. This process helps preserve important information while reducing the size of the impression. [13]

2.1.2.1.1.5 Fully Connected Layer

After passing through several convolutional and pooling layers, the feature maps are converted into a 1D vector. This vector is fed into one or more fully connected (FC) layers, where each node is connected to every node in the previous layer. These layers extract further widths using the extracted features. Output: A vector representing learned features from the entire image. [13]

2.1.2.1.1.6 Output Layer

The output layer often employes a softmax function (for classification) to generate a probability distribution across the possible classes.

The matematical insights:

$$Softmax(z_i) = \frac{e^{z_i}}{\sum_j e^{z_j}} \tag{2.4}$$

Where z_i is the score for class i and j represents all classes. [13]

2.1.2.2 R-CNN Region Based Convolutional Neural Networks

R-CNN is a pioneering architecture in object detection. It effectively bridges the gap between deep learning and object detection, offering significantly improved accuracy over traditional methods by leveraging region-based approaches.

`

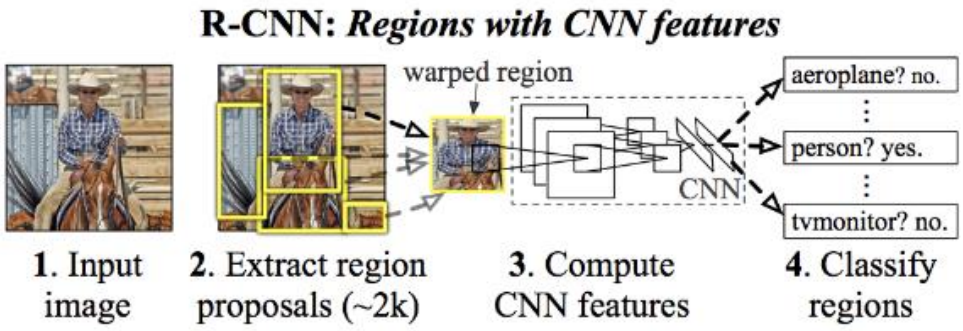


Figure 2: R-CNN Architecture

2.1.2.2.1 Input Image and Region Proposal Generation

R-CNN focuses on specific areas called region proposals. Region proposals are generated using external algorithms like Selective Search, which segment the image into ~2000 candidate regions likely to contain objects.

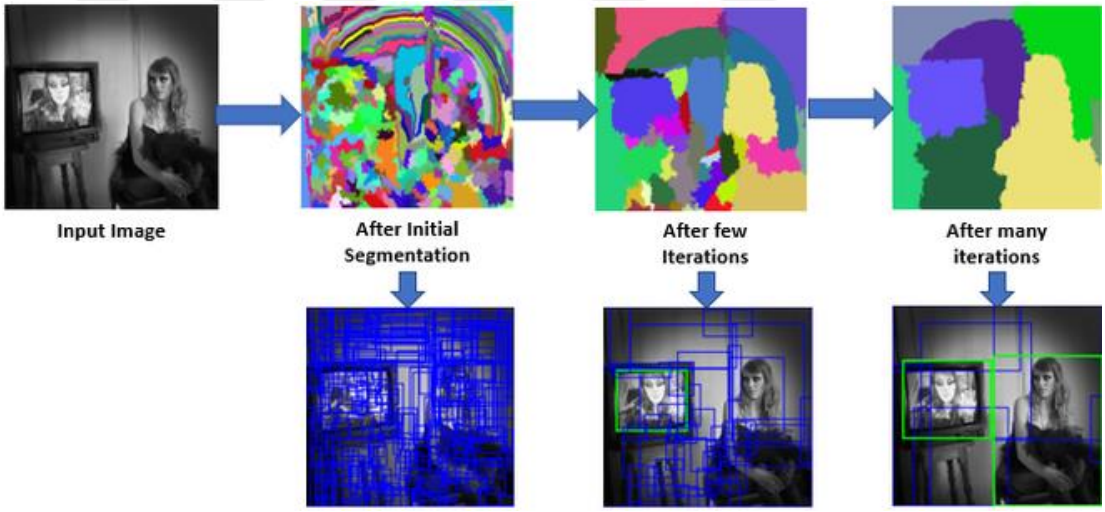


Figure 3: Region Proposal Method Examples

These proposals are determined based on low-level image features such as: Color similarity, Texture consistency, Size, Shape compatibility.

The output is a list of bounding box coordinates [x1,y1,x2,y2] representing rectangular regions of interest (ROIs). This step drastically reduces the computational burden by narrowing down the focus to a small subset of regions. [4]

2.1.2.2.2 Feature Extraction for Each Region

Each region proposal is subsequently cropped from the original image and reduced to a fixed dimension (e.g., 224×224) to match the input size required by a

`

CNN. These resized areas are passed through a pre-trained convolutional neural network, e.g., AlexNet or VGGNet, to obtain high-quality feature representations.

- **Low-level features capture:** edges, textures, and basic patterns.
- **High-level features encode:** more abstract concepts such as object parts and semantic meanings.

CNN generates a feature vector for each region proposal, which captures visual data of the region. This feature vector is the basis for classification and bounding box refinement. [4]

2.1.2.2.3 Classification and Bounding Box Refinement

The feature vectors extracted by the CNN are passed through two key components:

- **Class-specific Support Vector Machines (SVMs):** Each SVM is responsible for determining whether a region belongs to a specific object class (e.g., "knife," "gun," or "bottle") or is background.
- **Bounding Box Regression:** To improve the localization accuracy, R-CNN applies a regression model. This model adjusts the bounding box coordinates to better align with the ground-truth object boundaries.

The regression uses a mathematical model to minimize the difference between calculated and real coordinates:

$$\Delta x = \omega * (x_{gt} - x_{pred}) \quad (2.5)$$

Where x_{gt}, y_{gt} are ground truth coordinates and w ,h a regression weights. [4]

2.1.2.2.4 Final Output

After classification and bounding box regression, the system produces the final output; Object labels for each detected object (e.g., "knife," "bottle"). Refined bounding boxes that accurately encapsulate the objects. [4]

`

2.1.2.2.5 Strengths of R-CNN

- **Accurate Feature Representation**: By leveraging CNNs, R-CNN produces high-quality features compared to traditional hand-crafted methods.
- **Focus on Promising Regions**: Instead of evaluating every pixel, R-CNN focuses computation on region proposals, making it more efficient than exhaustive search methods.
- **Strong Performance**: Achieved a mean Average Precision (mAP) of 53.3% on the PASCAL VOC 2012 dataset, setting a new benchmark for object detection at its time. [4]

2.1.2.2.6 Limitations of R-CNN

- **Slow Inference**: Processing each region proposal independently through the CNN is computationally expensive, making real-time applications challenging.
- **Storage Requirements**: Features for all region proposals are stored, consuming significant disk space.
- **Non-End-to-End Training**: The multi-stage training process (CNN, SVMs, and regression) complicates optimization and integration. [4]

2.1.2.3 Fast R-CNN: Faster and Unified Object Detection

Fast R-CNN, introduced by Girshick in 2015, builds upon the foundations of R-CNN with significant improvements in speed and efficiency. By adopting a more streamlined architecture, it addresses the computational bottlenecks and storage limitations of its predecessor while maintaining high detection accuracy.

`

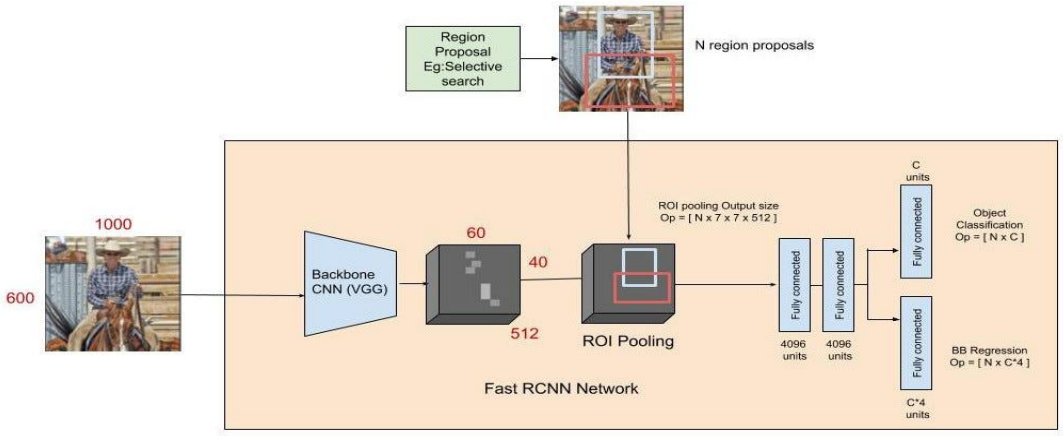


Figure 4: Fast R-CNN Architecture

2.1.2.3.1 Input Image and Shared Convolutional Feature Extraction

While R-CNN handles each region proposal individually, Fast R-CNN processes the entire input image in one forward pass. The input image is fed into a convolutional neural network (e.g., VGG16 or ResNet) to generate a shared feature map. This feature map includes spatially rich seam of information that represents the entire image. [14]

2.1.2.3.2 Region Proposal Projection

Instead of cropping the image into region proposals, Fast R-CNN directly projects the region proposals (ROIs) onto the shared feature map. These proposals are generated by external algorithms such as Selective Search, similar to R-CNN.

Each ROI is represented as a bounding box with coordinates [x1,y1,x2,y2].These coordinates define a rectangular region in the feature map corresponding to the region proposal in the original image. [14]

2.1.2.3.3 ROI Pooling Layer

To ensure consistent input dimensions for the fully connected layers, each ROI is resized to a fixed size using an ROI Pooling layer. ROI Pooling divides the projected region into a fixed grid (e.g., 7×7) and applies max pooling to each grid cell. This operation extracts a fixed-size feature map for each region, regardless of the original region's size. [14]

`

2.1.2.3.4 Feature Vector Extraction

The fixed-size feature maps are flattened into feature vectors. These vectors capture both spatial and semantic information from the original image regions. These feature vectors are passed through fully connected layers to enable classification and bounding box regression. [14]

2.1.2.3.5 Simultaneous Object Classification and Bounding Box Refinement

Fast R-CNN optimizes both object classification and bounding box regression in a single network, making it an end-to-end trainable system:

- **Classification Head**: A softmax layer predicts the class probabilities for each ROI, including a "background" class for non-object regions.
- **Bounding Box Regression Head**: A regression layer adjusts the bounding box coordinates to improve localization accuracy.

Both outputs are generated simultaneously, significantly reducing computational overhead. [14]

2.1.2.3.6 Final Output

- **Object Labels**: Predicted classes for each ROI (e.g., "knife," "gun").
- **Refined Bounding Boxes**: Adjusted coordinates for precise object localization. [14]

2.1.2.3.7 Strengths of Fast R-CNN

- **End-to-End Training:** All components (feature extraction, classification, and regression) are trained simultaneously, simplifying optimization.
- **Faster Inference:** Shared computation on the feature map drastically reduces the number of CNN forward passes (one instead of thousands).
- **Reduced Storage Requirements:** Features are computed on-the-fly rather than pre-computed and stored for each region, saving significant disk space. [14]

`

2.1.2.3.8 Limitations of Fast R-CNN

- **Dependence on External Region Proposal Algorithms:** Region proposals are still generated using external methods like Selective Search, which are computationally expensive.
- **Real-Time Processing:** While faster than R-CNN, Fast R-CNN is not suitable for real-time applications, as it still relies on slow region proposal generation. [14]

2.1.2.4 Faster R-CNN

Faster R-CNN, introduced by Ren et al. in 2015, builds upon Fast R-CNN by addressing the computational bottleneck of external region proposal generation. It introduces the **Region Proposal Network (RPN)**, which makes the system nearly end-to-end trainable and significantly faster.

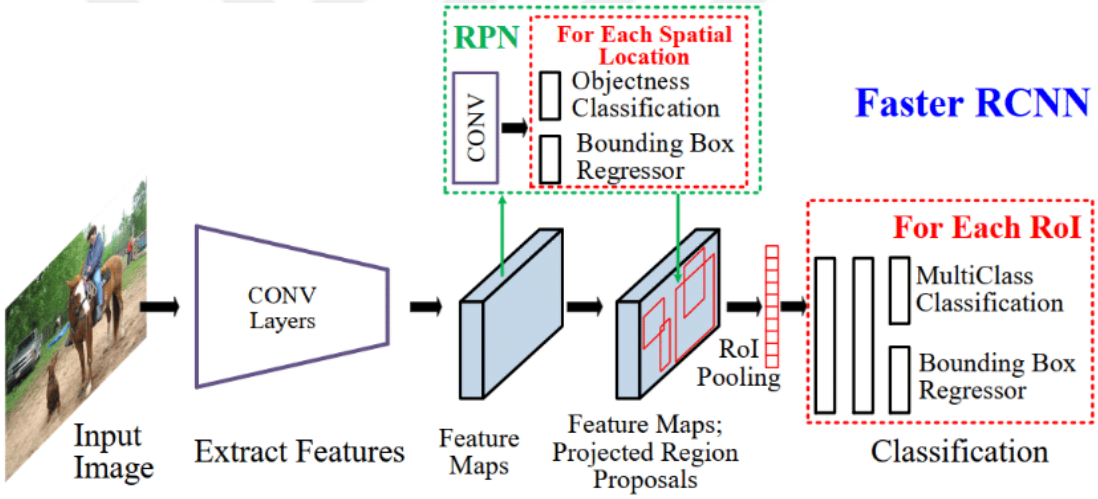


Figure 5: Faster R-CNN Architecture

2.1.2.4.1 Input Image and Shared Feature Extraction

The input image (e.g., baggage X-ray image) is passed through a convolutional neural network (**e.g., ResNet or VGG16**) to generate a shared feature map. This feature map is used for both region proposal generation and subsequent classification tasks. [15]

2.1.2.4.2 Region Proposal Network (RPN)

Faster R-CNN introduces the RPN to replace external algorithms (like Selective Search) for region proposal generation. The RPN is a lightweight CNN that scans the shared feature map to generate region proposals.

`

Anchor Boxes: At each feature map location, RPN generates multiple anchor boxes (e.g., 9 anchors of varying scales and aspect ratios).

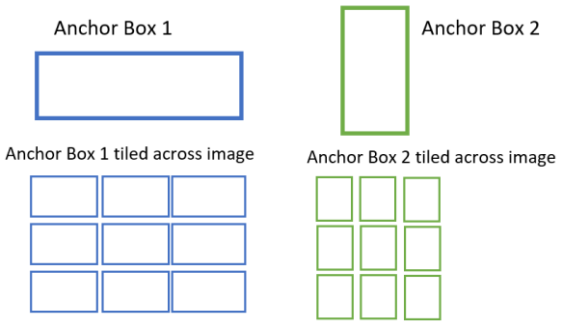


Figure 6: Anchor Boxes

Each anchor box is evaluated as either an object or background.

Binary Classification: A small classification layer predicts whether each anchor contains an object or is background.

Bounding Box Regression: A regression layer adjusts anchor box coordinates to better fit the object.

Output: A set of refined region proposals (ROIs) represented as bounding box coordinates [x1,y1,x2,y2] with associated objectness scores. [15]

2.1.2.4.3 ROI Pooling (or ROI Align)

The proposed regions are projected onto the shared feature map and resized to a fixed size using ROI Pooling or ROI Align (for better spatial precision). Fixed-size feature maps (e.g., 7×7) are extracted for each region.

$$RoI\ Align(x,y) = \sum_i \sum_j Feature[i,j] * BilinearInterpolation(x,y) \quad (2.6)$$

Where i,j are the grid cells, and x,y are the coordinates in the feature map. [15]

2.1.2.4.4 Feature Vector Extraction

Fully connected layers process the fixed-size feature maps to create feature vectors, which serve as inputs for classification and regression. [15]

`

2.1.2.4.5 Simultaneous Object Classification and Bounding Box Refinement

- **Classification Head:** A softmax layer predicts class probabilities for each ROI.
- **Bounding Box Regression Head:** A regression layer further adjusts the coordinates of each ROI for better localization. [15]

2.1.2.4.6 Final Output

- **Object Labels:** Predicted classes for each ROI (e.g., "knife," "bottle").
- **Refined Bounding Boxes:** Adjusted coordinates for precise localization of detected objects. [15]

2.1.2.4.7 Strengths of Faster R-CNN

- **Integrated Region Proposal Generation:** The RPN eliminates the need for slow external algorithms like Selective Search, making the system nearly end-to-end trainable.
- **Faster and More Efficient:** By sharing convolutional computation between the RPN and the detection network, Faster R-CNN achieves significantly faster inference without sacrificing accuracy.
- **High Accuracy:** Faster R-CNN achieves state-of-the-art performance on multiple object detection benchmarks (e.g., PASCAL VOC, MS COCO). [15]

2.1.2.4.8 Limitations of Faster R-CNN

- **Not Real-Time:** Despite improvements, depending on the computational complexity, Faster R-CNN is not suitable for real-time applications.
- **Anchor Design Dependency:** The performance heavily depends on the choice of anchor box sizes, scales, and aspect ratios, requiring careful tuning. [15]

2.1.2.5 Single Shot MultiBox Detector (SSD)

The Single Shot MultiBox Detector (SSD) is a state-of-the-art object detection framework that can detect objects in images and return bounding boxes and class labels

`

with high accuracy. SSD is known for its efficiency and high-speed performance. Unlike earlier object detection methods like R-CNN and its variants, SSD performs detection in a single forward pass through the network.

2.1.2.5.1 Introduction

SSD is an object detection algorithm that detects multiple objects in a single pass of the network, which gives it the advantage of real-time performance. SSD is based on Convolutional Neural Networks (CNN) and utilizes both high-level and low-level feature maps for object detection. SSD performs detection by predicting the offsets for bounding boxes and the corresponding object class probabilities. [5]

2.1.2.5.2 SSD Architecture

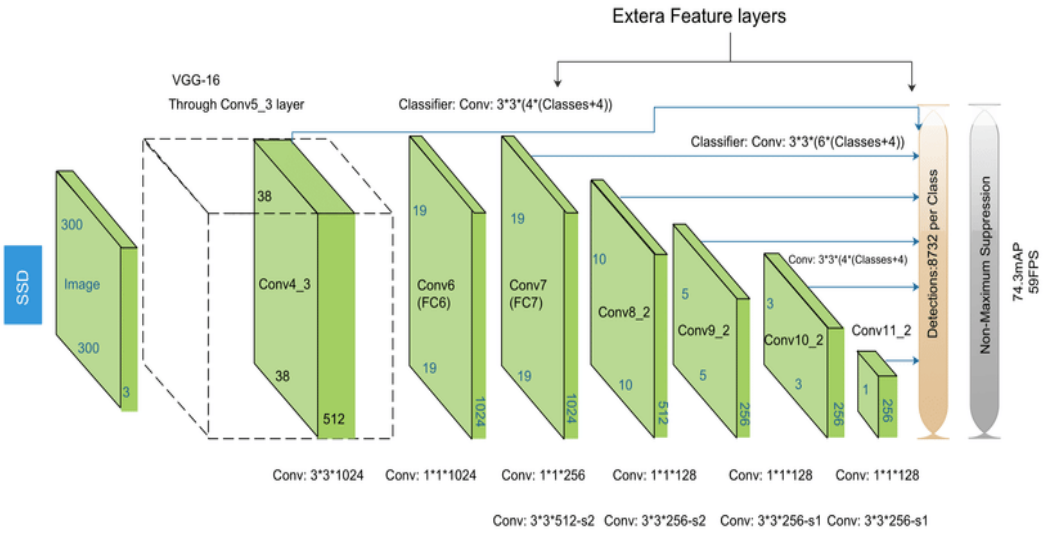


Figure 7: SSD Architecture

2.1.2.5.2.1 Input Image

The input image is usually resized to a fixed size (e.g., 300x300 pixels) for compatibility with the network.

The image is passed through a base CNN, which produces feature maps. [5]

2.1.2.5.2.2 Base Network (e.g., VGG16 or ResNet)

A pre-trained CNN (e.g., VGG16 or ResNet) serves as the base network for feature extraction.

`

The base network processes the input image and produces feature maps at various levels of abstraction (low- and high-level features). [5]

2.1.2.5.2.3 Multiscale Feature Maps

SSD extracts feature maps from several layers of the base network. These feature maps are of different sizes, allowing the network to detect objects of varying sizes. [5]

2.1.2.5.2.4 Convolutional Detection Layers

SSD applies a set of convolutional layers at each scale of feature maps to predict bounding box offsets and class scores. For each location on the feature map, SSD predicts, **Bounding box** offsets (relative to the default box), **Class scores** (probabilities for each object class). [5]

2.1.2.5.2.5 Bounding Box Prediction

SSD predicts the offsets for bounding boxes relative to default (anchor) boxes at each location on the feature map. These predictions consist of, coordinates of the bounding box (center, width, and height), Confidence scores for each class. [5]

2.1.2.5.2.6 Non-Maximum Suppression (NMS)

After the bounding box predictions are made, Non-Maximum Suppression (NMS) is applied to remove duplicate or highly overlapping bounding boxes. NMS selects the box with the highest confidence score and discards other boxes with high Intersection over Union (IoU) values. [5]

$$IoU(A, B) = \frac{Area\ of\ Intersection}{Area\ of\ Union} \tag{2.7}$$

2.1.2.5.2.7 Final Output

The final output consists of bounding boxes with class labels and confidence scores for each detected object. [5]

`

2.1.2.5.3 Strengths of SSD

- **Speed:** SSD is significantly faster than region-based methods like Faster R-CNN because it does not use a region proposal network (RPN) and performs object detection in a single pass.
- **Real-time Detection:** SSD can process images in real-time, making it suitable for applications like video analysis.
- **Multiscale Detection:** The use of feature maps from multiple scales enables SSD to detect objects of varying sizes.
- **Efficiency:** SSD reduces the need for exhaustive search over image regions, making it computationally efficient. [5]

2.1.2.5.4 Limitations of SSD

- **Accuracy:** SSD tends to have lower accuracy compared to two-stage detectors like Faster R-CNN, especially for small objects.
- **Complexity in Implementation:** While faster, SSD's implementation is more complex compared to simpler classifiers like CNNs for image classification.
- **Bounding Box Prediction:** SSD may struggle with precise bounding box localization for complex or overlapping objects. [5]

2.1.2.6 You Only Look Once

YOLO is a popular and highly efficient deep learning model for real-time object detection. Unlike traditional object detection methods (such as R-CNN), YOLO is a single-stage detector that predicts both the class labels and the bounding boxes for objects in a single pass through the network. This makes it extremely fast and suitable for real-time applications.

2.1.2.6.1 Introduction

YOLO is an end-to-end deep learning algorithm that can simultaneously predict the location (bounding box) and class of objects in an image in real-time. Unlike traditional detection methods, which are based on region proposals (e.g., Faster R-CNN), YOLO divides the image into a grid and predicts bounding boxes and class probabilities for each grid cell. [6]

`

2.1.2.6.2 Key Features of YOLO

- **Real-Time Speed:** YOLO is known for its fast inference time, making it suitable for applications requiring real-time object detection.
- **Unified Architecture:** The entire detection pipeline is trained end-to-end, making it easier to implement and optimize compared to multi-stage methods.
- **Global Context:** YOLO looks at the entire image when making predictions, thus considering contextual information across the whole scene rather than focusing on individual object regions. [6]

2.1.2.6.3 YOLO Architecture

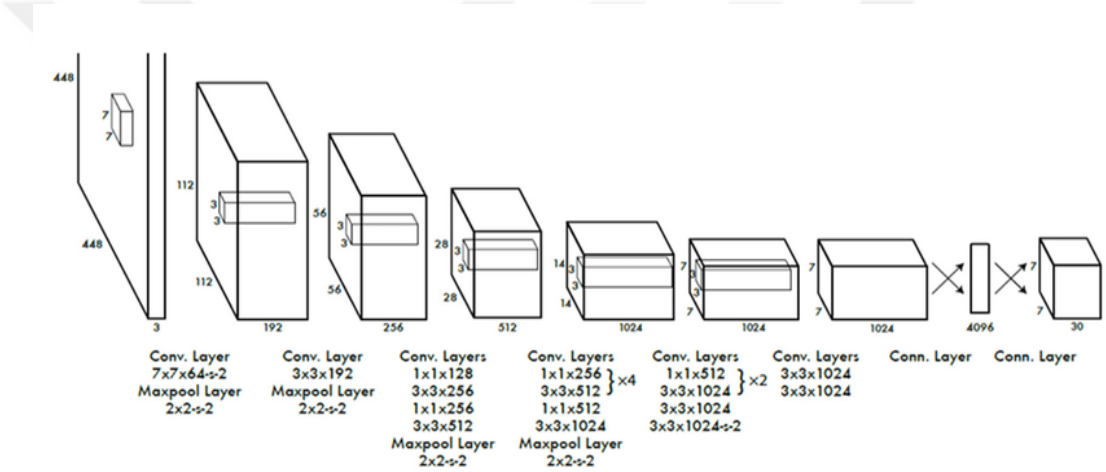


Figure 8: Yolo Architecture

2.1.2.6.3.1 Input Image

The input image is resized to a fixed size (e.g., 416x416 or 608x608 pixels). This step ensures that the image size matches the input size expected by the network. The image is then normalized (e.g., pixel values are scaled to the range [0, 1]). [6]

2.1.2.6.3.2 Feature Extraction

YOLO uses a CNN to extract features from the input image. The CNN is composed of several convolutional layers, pooling layers, and fully connected layers. For example, Darknet (a lightweight architecture used for YOLO) is used as the backbone for feature extraction. These features capture patterns such as edges, textures, shapes, and higher-level object parts. [6]

`

2.1.2.6.3.3 Grid Division

YOLO divides the image into a grid of size S×SS \times SS×S (for example, 13x13 or 19x19). Each grid cell is responsible for detecting objects whose center falls within that grid cell. For each grid cell, YOLO predicts bounding boxes and class scores. [6]

2.1.2.6.3.4 Bounding Box and Class Prediction

For each grid cell, YOLO predicts:

Bounding Box Coordinates: The location of the object relative to the grid cell (center, width, height).

$$L_{bbox} = \sum_{i} \sum_{j} 1_{obj} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 + (w_i - \hat{w}_i)^2 + (h_i - \hat{h}_i)^2 \right] \quad (2.8)$$

x_i, y, w_i, h_i Predicted bounding box center coordinates and dimensions

$\hat{x}_i, \hat{y}_i, \hat{w}_i, \hat{h}_i$ Ground truth bouding box values

1_{obj} Indicator function (1 if the grid cell contains an object, 0 otherwise) [6]

Confidence Score: The probability that the grid cell contains an object, and how accurate the predicted bounding box is.

$$C = P(object) * IoU_{pred,truth} \quad (2.9)$$

P(object) is the probability that the cell contains an object

$IoU_{pred,truth}$ is the intersection over union between the predicted and ground-truth bouding boxes. [6]

Class Probabilities: The likelihood that the object belongs to a particular class (e.g., car, person, dog). [6]

$$P(class_i) = \frac{\exp(z_i)}{\sum_j \exp(z_j)} \quad (2.10)$$

2.1.2.6.3.5 Non-Maximum Suppression (NMS)

After YOLO generates multiple bounding box predictions for each grid cell, NMS is applied to remove redundant bounding boxes. Ranking the bounding boxes

`

based on their confidence scores. Removing boxes that overlap with higher-confidence boxes, based on the IoU (Intersection over Union) threshold. [6]

2.1.2.6.3.6 Final Output

The final output is a list of bounding boxes along with their associated class labels and confidence scores for each detected object. [6]

2.1.2.6.4 Advantages of YOLO

- **Real-time Performance:** YOLO's architecture allows for real-time object detection, making it ideal for applications requiring fast response times.
- **End-to-End Training:** YOLO trains the entire model in a single step, simplifying the training process compared to multi-stage detectors like Faster R-CNN.
- **Global Context:** YOLO processes the entire image in one forward pass, considering the global context rather than focusing on local regions. [6]

2.1.2.6.5 Limitations of YOLO

- **Small Object Detection:** YOLO can struggle with detecting very small objects in an image due to the grid-based nature of its prediction.
- **Localization Accuracy:** YOLO's bounding box predictions may be less accurate compared to region-based methods, especially for small objects.
- **Class Imbalance:** If the dataset contains a high imbalance in object classes, YOLO might be biased toward predicting the more frequent classes. [6]

2.1.2.7 YOLO-v8 Advancements in Object Detection

2.1.2.7.1 Introduction

Building on the strong foundation of previous YOLO models, YOLOv8 presents cutting-edge developments in the structure and training of neural networks methodologies. Similar to its predecessors, YOLOv8 combines object detection and

`

classification into a seamless, end-to-end framework that achieves an optimal balance between performance speed and accuracy.[16]

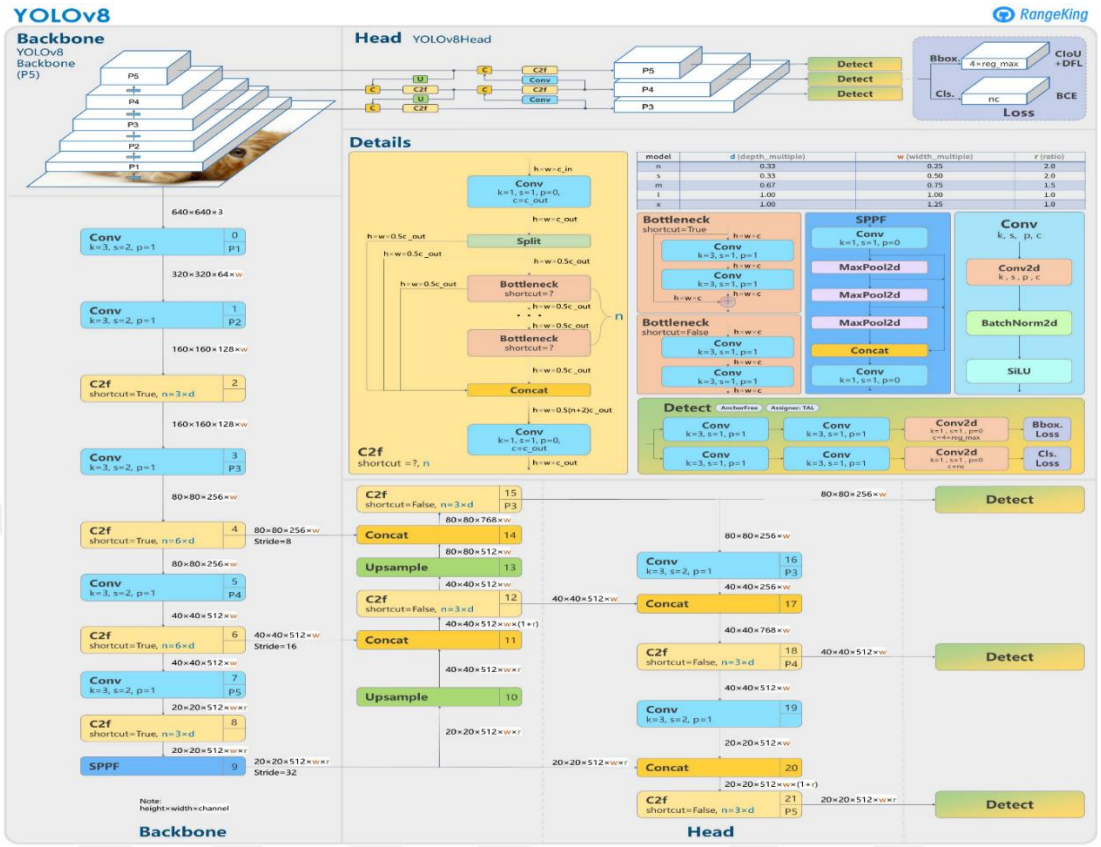


Figure 9: YOLOV8 Architecture

2.1.2.7.2 Architectural Innovations

2.1.2.7.2.1 Backbone

YOLOv8 employs an advanced convolutional neural network (CNN) backbone, optimized for speed and accuracy. It extracts multi-scale hierarchical features essential for detecting objects of various sizes. By leveraging efficient layers like depthwise separable convolutions, it minimizes computational overhead while maintaining robust feature representation.

2.1.2.7.2.2 Neck

The neck utilizes an enhanced Path Aggregation Network (PANet) to fuse multi-scale features effectively. This integration supports accurate detection of objects with diverse scales and sizes while optimizing memory and computational efficiency.

`

2.1.2.7.2.3 Head

The anchor-free head module simplifies bounding box predictions by eliminating the need for predefined anchors. This approach reduces hyperparameter dependency, improves adaptability to various object shapes, and enhances detection efficiency. [16]

2.1.2.7.3 Training Methodologies

2.1.2.7.3.1 Advanced Data Augmentation

Techniques like mosaic and mixup augmentations expose the model to a broader range of object scales and spatial configurations, enhancing its generalization capabilities. [16]

2.1.2.7.3.2 Loss Functions

- **Focal Loss:** Solves the class imbalance by focusing on difficult examples, improving classification accuracy.
- **IoU Loss:** Enhances bounding box accuracy with greater localization.
- **Objectness Loss:** Aims at object-dense regions, improving detection as a whole performance. [16]

2.1.2.7.3.3 Mixed Precision Training

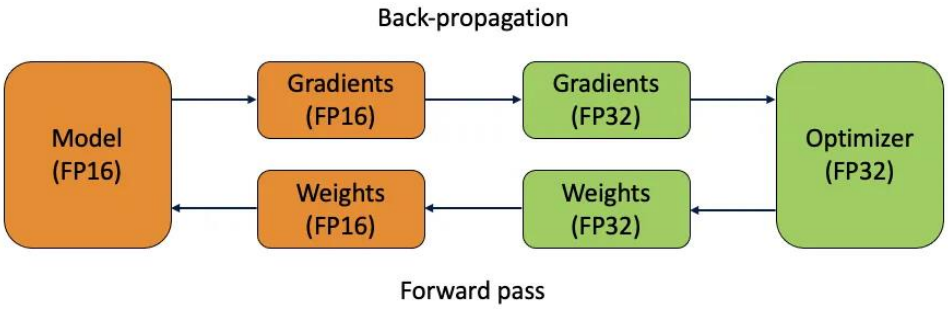


Figure 10: Mixed Precision Training Schema

Mixed precision training uses 16-bit floating-point computation, accelerating training on GPUs and reducing memory usage. This enables larger batch sizes and faster processing without sacrificing accuracy. [16]

`

2.1.2.7.3.4 Key Enhancements

- **CSP Backbone:** The Cross Stage Partial (CSP) bottleneck module reduces redundancy and improves feature reuse, enhancing efficiency and inference speed.
- **Enhanced PANet Neck:** The optimized PANet neck improves feature flow between the backbone and head, ensuring state-of-the-art performance in detecting small or densely packed objects.
- **Anchor-Free Prediction:** By departing from anchor-based methods, YOLOv8 reduces computational complexity and improves its ability to handle objects with varying scales and aspect ratios. [16]

2.1.2.8 YOLOv9: Advancements in Object Detection

2.1.2.8.1 Introduction

YOLOv9 builds on the strengths of its predecessors by incorporating innovative techniques to tackle the information bottleneck, further advancing accuracy and efficiency in object detection. While YOLOv8 emphasizes accuracy and speed optimization through enhanced PANet and CSPNet, YOLOv9 promote new two architectural components: Programmable Gradient Information (PGI) and the Generalized Efficient Layer Aggregation Network (GELAN). These innovations address the challenge of information loss during data flow through network layers, enhancing gradient stability and prediction accuracy. [17]

2.1.2.8.2 Architectural Footprint of YOLOv9

2.1.2.8.2.1 Backbone

For multi-scale feature extraction, YOLOv9 uses a CNN-based backbone similar to YOLOV8, enhanced by the integration of GELAN. GELAN is based on the Efficient Layer Aggregation Network (ELAN) using computational blocks including Resblocks and Darkblocks, CSP blocks. This design enables efficient feature extraction across network layers while preserving the basic hierarchical properties, and maintains the balance between accuracy and computational efficiency. [17]

2.1.2.8.2.2 Neck

YOLOv9's neck is built on the advancements of YOLOv8's PANet, and with the integration of PGI, it significantly improves the feature fusion process. YOLOv9

`

improves the fusion of features across layers by using multi-level auxiliary information from PGI, effectively reducing information loss during data flow through the network. This improvement makes YOLOv9 excellent at detecting objects of various sizes by balancing gradient computations. [17]

2.1.2.8.2.3 Head

YOLOv9 is based on the anchor-free bounding box estimation technique, which includes reversible functions introduced by PGI. The reversible architecture guarantees the preservation of underlying data in both forward and backward passes, resulting in more reliable estimations with less computational demand. This approach increases both inference speed and accuracy, making it particularly suitable for real-time applications. [17]

2.1.2.8.3Training Methodologies and Innovations

2.1.2.8.3.1 Advanced Data Augmentation

Similar to YOLOv8, YOLOv9 uses the augmentation techniques of mosaic and mixup to enhance model generalization. Whereas, with the introduction of PGI, the network can utilize the augmented data in a more effective manner by preventing gradients from being propagated in an inefficient way through the layers, even via shallow or thin models. [17]

2.1.2.8.3.2 Loss Functions

The loss function for YOLOv9 is based on components of the previous version, e.g., focal loss for classification and IoU loss for localization, while introducing improvements through PGI. The reversible branch on the side in PGI gives more accurate gradient updates, which are important when optimizing objectless loss and enabling the network to effectively detect object regions in images. [17]

`

2.1.2.8.3.3 Mixed Precision Training

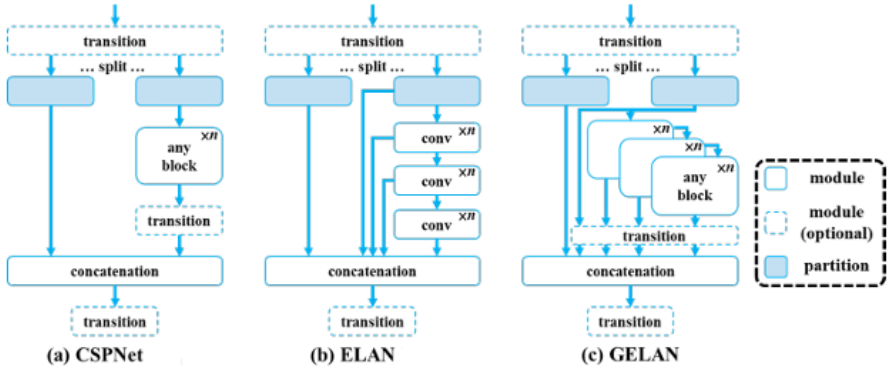


Figure 11: GELAN Structure

Like YOLOv8, YOLOv9 employs mixed precision training in order to speed up coherent GPUs. YOLOv9 integrates the GELAN framework, though, and goes a step further with efficiency. In its optimization of the utilization of computational blocks, GELAN reduces memory usage while maintaining high accuracy during inference, especially for light models. [17]

2.1.2.8.4 Revolutionary Techniques in YOLOv9

2.1.2.8.4.1 Programmable Gradient Information (PGI)

YOLOv9 features PGI, a groundbreaking design that utilizes an auxiliary reversible branch to generate consistent gradients while maintaining the same inference cost. By merging multi-level auxiliary information, this architecture overcomes the challenges of information loss often encountered in deep networks. This advancement markedly improves the performance of both compact and complex models during training. [17]

2.1.2.8.4.2 Generalized Efficient Layer Aggregation Network

GELAN, which was brought in YOLOv9, builds on the ELAN architecture of the previous versions with the addition of a range of computational blocks apart from standard convolutions. By combining the strengths of CSPNet and ELAN, GELAN realizes speedy inference without any loss of accuracy. Its adaptable structure makes the network efficient on diverse tasks and hardware, which makes it extremely appropriate for real-time object detection. [17]

`

2.1.2.8.5 Performance Analysis

The incorporation of PGI and GELAN in YOLOv9 is a dramatic departure from previous releases, opting to eliminate information bottlenecks instead as well as improving gradient stability. All of these are for the purpose of preserving vital information not lost through the layers of the network and even facilitating the development of optimal models that are equally, or better, as effective as their more voluminous versions. YOLOv9 has state-of-the-art performance across a variety of model sizes and deployment platforms, such as:

- **49% reduction** in parameters,
- **43% reduction** in computational cost, and
- **0.6% improvement** in accuracy on the MS COCO dataset. [17]

2.1.2.9 Yolov10

2.1.2.9.1 Real-time Object Detectors

Real-time object detection plays a pivotal role in real-world applications by ensuring low-latency performance. The YOLO series has significantly contributed to this domain with continuous improvements in architecture and performance. Early YOLO versions (YOLOv1, YOLOv2, YOLOv3) established a robust detection framework consisting of backbone, neck, and head components. YOLOv4 and YOLOv5 further enhanced performance with CSPNet design, data augmentation, and scalable model variants. YOLOv6 and later versions introduced advanced innovations such as BiC, SimCSPSPPF, E-ELAN, and C2f blocks to optimize feature extraction, gradient flow, and multi-scale feature fusion. YOLOv9 added GELAN and PGI to further refine the architecture and training processes. [18]

2.1.2.9.2End-to-End Object Detectors

End-to-end detection approaches, such as DETR, utilize transformer architectures to eliminate traditional components like NMS, achieving a direct prediction-to-output pipeline. Various improvements to DETR, including Deformable-DETR and DINO, have accelerated convergence speeds and reduced computation costs. Other CNN-based approaches, such as Learnable NMS and DeFCN, also work toward eliminating redundancy and optimizing the detection process for greater efficiency. [18]

`

2.1.2.9.3 Consistent Dual Assignments for NMS-free Training

This section introduces a dual label assignment strategy for YOLO models that removes the need for Non-Maximum Suppression (NMS) during inference while improving performance. [18]

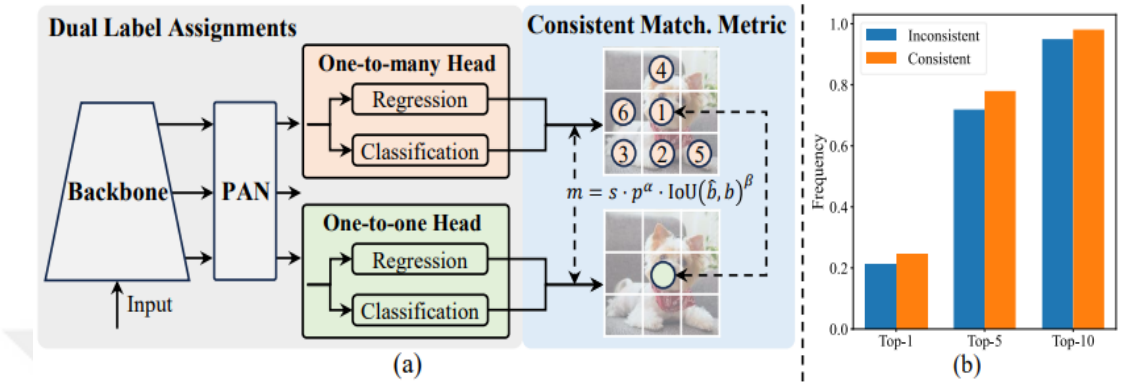


Figure 12: NMS Free Training

2.1.2.9.4 Dual Label Assignments

One-to-many assignment provides strong supervisory signals for training but requires NMS to eliminate redundant predictions, impacting inference efficiency.

One-to-one matching avoids redundancy but offers weaker supervision, slowing convergence and affecting accuracy.

The proposed method combines both assignments into a unified model, with one branch using one-to-many assignment for strong supervision and the other utilizing one-to-one matching to avoid NMS during inference. This method results in faster inference without sacrificing accuracy. [18]

2.1.2.9.5 Inference Process

Both branches are optimized together during training. Inference uses the one-to-one branch, eliminating post-processing and increasing inference speed. [18]

2.1.2.9.6 Consistent Matching Metric

A uniform matching metric takes classification scores, IoU, and spatial priors. Hyper-parameters are shared between branches, making optimization easier and performance. [18]

`

2.1.2.9.7 Addressing Supervision Gaps

The supervision gap between supervision is reduced by a 1-Wasserstein distance in such a way that there is better consistency and leveraging more informative supervision in training. [18]

2.1.2.9.8 Performance and Comparison

The dual assignment technique outperforms existing methods like H-DETR and Group-DETR with better efficiency and accuracy without additional inference cost or tuning hyperparameters. [18]

2.1.2.9.9 Efficiency-Accuracy Driven Model Design

This section outlines the strategies used to balance computational efficiency and detection accuracy in YOLO architectures. [18]

2.1.2.9.9.1 Efficiency-Driven Design

- **Lightweight Classification Head:** A more efficient classification head reduces computational costs without compromising performance.
- **Spatial-Channel Decoupled Downsampling:** Separated downsampling and channel transformation reduce computation and parameters, improving efficiency.
- **Rank-Guided Block Design:** Compact blocks replace redundant layers in deeper stages, optimizing the architecture without sacrificing accuracy. [18]

2.1.2.9.9.2 Accuracy-Driven Design

- **Large-Kernel Convolutions:** Selective use of large kernels in deeper layers enhances global context capture while maintaining smaller object detection.
- **Partial Self-Attention (PSA):** By applying self-attention selectively to parts of feature maps, PSA reduces computational complexity and memory usage. [18]

`

2.1.2.9.10 Overall Strategy

With the application of lightweight components and decoupling of tasks that have been traditionally fused for a long time, YOLOv10 model ensures high performance at low computational overhead. Such development is critical in real-time systems where speed is equated to accuracy and low latency and quick inference in complex tasks.

2.2 APPLICATIONS IN X-RAY SECURITY

X-ray security image object detection has proven to be a tricky task caused by complex backgrounds, occlusions of objects, and overlap of instances. Various research has shown ways in which detection performance and accuracy can be enhanced.

Zhang et al. [19] presented the YOLOv5s-SE algorithm to solve the problems of X-ray security images. By the addition of a personalized channel attention module, the algorithm tends to pay attention to feature channels with information and suppresses those with less information. The GIoU loss was substituted with CIoU loss, which resulted in faster convergence. Experiments showed precision, recall, and mAP improvements over YOLOv5s and other approaches.

Chen et al. [20] proposed a Recursive CNN model with a multichannel region proposal network (MCRPN) to enhance anomaly detection in X-ray security images. By fusing high-level semantic components and low-level edge features from the VGG16 network, their method achieved an average detection accuracy of 84.69% on the SIXray OD dataset, outperforming the baseline by 6.28%.

Another significant contribution is by Singh et al. [21], who explored various end-to-end object detection architectures, including Faster R-CNN, Mask R-CNN, and RetinaNet, for firearm detection. Their evaluation demonstrated that Faster R-CNN with ResNet101 achieved the highest performance, with mAP values of 0.91 and 0.88 for a two-class detection task across different X-ray datasets.

Das et al. [22] evaluated the applicability of traditional sliding window CNNs and region-based methods like Faster R-CNN and R-FCN for detecting objects in X-ray baggage imagery. Their results showed that Faster R-CNN with VGG16, pretrained on ImageNet, achieved superior performance with an 88.3% mAP for a six-class detection problem.

`

Chen et al. [23] presented a novel object detection approach using Faster R-CNN for X-ray baggage security imagery. Their framework incorporated a foreground-background segmentation method and utilized transfer learning to enhance the model's efficiency, leading to improved detection results.

In order to counteract the shortage of annotated data, Huang et al. [24] proposed a means of constructing realistic synthetic datasets. It helped in the simulation of nocclusions, multiple textures, and distractors to a significant extent, greatly contributing towards the training of deep networks. Experiments ensured the supremacy of synthetic datasets over human-annotated datasets for X-ray security screening.

Kumar et al. [25] made a comparison among different object detection models like Faster R-CNN, SSD, and YOLOv3, on single-channel (GDXray) and multichannel (SIXray) datasets. They reported the efficiency of Faster R-CNN with ResNet50, with a 0.966 mAP for four-class detection on GDXray and a 0.845 mAP for two class detection on SIXray.

Singh and Kaur [26] examined CNN-based techniques like YOLO and Faster R-CNN for threat detection. Using Faster R-CNN with ResNet, pretrained on ImageNet, they achieved an accuracy of 98.4% for four-class threat recognition with minimal computational overhead.

Yang et al. [27] proposed an X-ray security inspection algorithm based on an improved YOLOv4 model. By integrating deformable convolution, GHM loss, and a hybrid non-maximum suppression method combining soft NMS and DIoU NMS, the algorithm achieved a significant 91.4% mAP, outperforming YOLOv4 by 3.3% while meeting real-time processing requirements.

Liu et al. [28] introduced the PIDray dataset, a comprehensive collection of 47,677 X-ray images annotated with segmentation masks and bounding boxes, covering 12 categories of prohibited items. This dataset is one of the largest of its kind, supporting robust model development. They also proposed the Selective Dense Attention Network (SDANet), comprising dense attention and dependency refinement modules, providing a strong baseline for detection tasks.

Zhang et al. [29] developed a novel image synthesis method for hazardous object detection in X-ray imagery. By estimating difficult-to-detect positions, their method generates a difficulty map, enabling improved training data generation, which

`

enhances the performance of state-of-the-art object detectors compared to conventional random synthesis methods.

Wang et al. [30] combined Convolutional Neural Networks (CNN) with Long Short-Term Memory (LSTM) to generate real-time captions for X-ray images. Their model detected keywords such as “Gun” and “Knife,” triggering alerts for security personnel. The GDXray dataset was utilized to effectively train the model.

Chen et al. [31] proposed a YOLO-based framework to address occlusion and recognition challenges in X-ray imagery. They explored both transfer learning and training from scratch using the IEDXray dataset, achieving robust results in detecting improvised explosive devices.

Li et al. [32] employed transfer learning approaches to optimize pre-trained CNNs for object classification in X-ray baggage screening. By fine-tuning models initially trained on generalized datasets, their approach significantly improved classification performance, even with limited domain-specific data.

Huang et al. [33] studied the use of high and low X-ray energy response imagery for joint object detection and segmentation tasks. They evaluated architectures such as Mask R-CNN and YOLOv2, demonstrating that raw variant imagery can improve detection accuracy across different scanner configurations.

Zhao et al. [34] utilized CNN-based approaches for automatic threat detection in security X-ray images. By leveraging the Passenger Baggage Object Database (PBOD), their models achieved high reliability, with heatmaps introduced to visualize threat locations, and an AUC of 0.95 on the ROC curve.

Wang et al. [35] explored the applicability of Bag of Visual Words (BoW) methods for classifying and retrieving X-ray images. They leveraged the unique properties of X-ray imagery, enhancing performance and improving upon traditional BoW applications.

Yang et al. [36] proposed a multi-scale CNN for material classification in dual-energy X-ray images. Their architecture balanced receptive fields and detail preservation, outperforming traditional methods in classifying materials like metals and organics.

Jiang et al. [37] X-ray imagery applications in fine-tuned pre-trained CNNs, surpassing traditional feature extraction methods. They evaluated models like YOLOv2 and Faster R-CNN, achieving high mAP values and proving effective for complex object localization tasks.

`

Zhang et al. [38] leveraged terahertz (THz) technology to develop a two-stage classifier for detecting concealed and visible dangerous equipment. Their system demonstrated promising results in classifying objects in both natural and THz images.

Zhao et al. [39] employed single-stage detectors such as SSD and RetinaNet to detect sharp items in X-ray images. Their approach proved effective for real-time detection tasks, addressing challenges in cluttered security imagery.

Liu et al. [40] introduced a Label-Aware (LA) mechanism to improve the detection of overlapping objects in X-ray security imagery. The method demonstrated superior accuracy, validated on datasets like OPIXray and CLCXray.

Wang et al. [41] evaluated various computer vision strategies, including deep learning and classic pattern recognition, for X-ray testing in baggage inspection. Their experiments using the GDXray database revealed the strengths and limitations of different approaches for recognizing threat objects such as handguns and razor blades.

2.3 CHALLENGES IN X-RAY IMAGING

X-ray imaging is one of security's most used applications that is mainly applied in airport baggage scanning. Despite its popularity, there is a list of challenges that are encountered in applying X-ray imaging, more in object detection using it. The majority of these challenges arise from overlapping objects, noise, challenges in a particular domain, and imaging technologies in general limitations. In this section, various of the most critical challenges that affect detection systems using X-ray imaging, more in object detection in security scenarios are discussed.

2.3.1 Overlapping Objects

Overlapping or occluded objects constitute one of the most serious challenges in X-ray imaging. In real baggage screening, objects overlap or lie side to side, resulting in object occlusion in half or in entirety in the X-ray imaging. This hampers object identification and object localization, especially when objects consist of neighboring densities or material properties. Overlap also results in missed detection or alarms.

Impact on Detection: The object detection and accurate object localization is a fundamental process of security screenings. Occlusion hampers human inspectors and automation systems in object identification and object separation in complex imaging [42],[43].

`

2.3.2 Noise and Artifacts

X-ray images get corrupted by artifacts and noise due to various reasons such as defects in imaging processes, material densities, or different X-ray beam intensities. The artifacts conceal object details in the picture, resulting in detection inaccuracies. Random or grainy pixel nature of noise can hamper object or even object class separation.

Impact on Detection: Noise and artifacts severely degrade image quality, resulting in detection error, especially in small or textured objects. Noise and artifacts also make object detection models more difficult to design reliably [44],[43].

2.3.3 Domain-Specific Issues

X-ray imaging systems used in security screenings generally function over a range of domain-specific issues. Such issues range from different luggage types to be screened, different arrangements of X-ray devices, to different material components in objects. All such variation results in different image quality and difficulty in designing a detection system that is uniform over different working settings.

Impact on Detection: In consistencies in object types, X-ray beam strength, and imaging parameters can lead to different object appearances in X-ray images. Such issues hamper one's capability of designing a detection model that is equally skilled in all settings [44].

2.3.4 Limited Ground Truth Annotations

Another challenge of imaging using X-rays is that there is a lack of ground truth labels. There are different X-ray datasets that are sparsely or inaccurately annotated, especially for objects that are partially visible or occluded.

They limit detection models during their training. Labels of objects that are occluded or just partially visible generally do not exist, resulting in poor model performance and inadequate training.

Impact on Training Models: Inaccurate or poor labels result in models getting trained on incomplete or deceptive data, having a deleterious effect on their object detection in practical applications [43],[44].

`

2.3.5 Summary

The challenges of sparsity of ground truth labels, overlapping objects, artifacts, and noise pose serious challenges to object detection system design based on X-ray imaging. The challenges affect detection precision and transferability, without which security screening is useless. In response to such challenges, there is a need to move ahead in preprocessing of data, in new architectures, and in more realistic and accurate datasets. Tackling these challenges is of great concern to advancing security systems based on X-ray imaging to be more reliable and efficient.

2.4 BENCHMARK DATASETS

In the use of airport security systems to detect objects via X-rays, various benchmark datasets have been developed to facilitate improvement in machine learning models. The object detection system is compared and improved in terms of detection of prohibited objects in luggage using such datasets as CLCXray, PIDXray, and SIXray. An introduction to each of these datasets is presented below in terms of their uniqueness, challenges, and scholarly contribution.

2.4.1 CLCXray Dataset

The CLCXray dataset is constructed in order to enhance object detection and classification in checked baggage screening. It is directed at detecting prohibited items in complex baggage arrangements commonly encountered in airport security.

2.4.1.1 Dataset Details

- Image Content: X-ray images of luggage containing various items, including prohibited objects.
- Prohibited Item Categories: Sharp objects (e.g., knives, scissors), liquids (e.g., bottles), and electronics (e.g., laptops, power banks).

2.4.1.2 Key Features

- Complex Overlaps: Objects often overlap or are occluded, making detection difficult.
- Realistic Scenarios: Reflects actual luggage configurations.
- Image Diversity: Includes varying orientations and object sizes.

`

2.4.1.3 Challenges

- Occlusion and Clutter: Items often overlap, complicating detection.
- Noise and Distortion: Variations in X-ray intensity due to material differences.

2.4.1.4 Significance

This dataset supports the development of machine learning models tailored for complex real-world airport security systems. [42]

2.4.2 PIDXray Dataset

The PIDXray dataset is geared towards personal luggage screening and provides a varied collection of X-ray images for banned item discovery typically carried in hand luggage.

2.4.2.1 Dataset Details

- Image Content: X-ray images of small personal items in hand luggage.
- Prohibited Item Categories: Guns, knives, liquid bottles, and suspicious items.

2.4.2.2 Key Features

- Small Object Detection: Focuses on detecting smaller objects concealed within larger objects.
- Real-World Complexity: Simulates real-world personal luggage setups.

2.4.2.3 Challenges

- Concealment: Items are often hidden among dense clusters of personal belongings.
- Scale Variability: Extreme size variations of objects across images.
- Blur and Noise: Image artifacts due to material interference.

2.4.2.4 Significance

The PIDXray dataset is important for developing techniques for identifying concealed prohibited items in carry-on luggage. [44]

`

2.4.3 SIXray Dataset

The SIXray dataset is one of the largest X-ray datasets used for security inspection, offering millions of X-ray images ideal for deep learning training and evaluation.

2.4.3.1 Dataset Details

- Image Content: Real-world luggage configurations containing various items.
- Prohibited Item Categories: Guns, knives, scissors, and other dangerous items.
- Benign Item Categories: Non-prohibited items commonly found in luggage (e.g., clothes, electronics).

2.4.3.2 Key Features

- Imbalanced Distribution: Prohibited items are underrepresented, reflecting real-world scenarios.
- High Diversity: Includes a variety of luggage types and item configurations.

2.4.3.3 Challenges

- Data Imbalance: The skewed distribution of prohibited and benign items poses challenges for training effective models.
- High Noise: Variations in materials and imaging artifacts.

2.4.3.4 Significance:

The SIXray dataset is crucial for developing large-scale security inspection systems, particularly for detecting rare and dangerous objects in cluttered environments.[43]

2.4.4 GDX-ray Dataset

The GDX-ray dataset is designed for the development and benchmarking of object detection algorithms, specifically for detecting items in baggage X-ray images.

`

This dataset includes a variety of objects commonly found in baggage and aims to improve the detection of both benign and hazardous items.

2.4.4.1 Dataset Details

- Image Content: X-ray images of various objects found in luggage, including bags, electronics, weapons, and everyday items.
- Object Categories: Weapons (guns, knives), electronics (laptops, phones), liquids, and miscellaneous objects typically found in baggage.

2.4.4.2 Key Features

- Large-Scale Dataset: Contains thousands of high-resolution X-ray images, providing a diverse set of data for training and evaluation of object detection models.
- Wide Variety of Objects: Covers a broad range of object types, making it suitable for general object detection as well as specific security screening applications.

2.4.4.3 Challenges

- Class Imbalance: Some object categories, such as hazardous items, may be underrepresented, leading to potential challenges during model training for rare object detection.
- Class Ambiguity: Some objects may appear similar or have similar shapes, which could lead to difficulties in distinguishing between categories.

2.4.4.4 Significance

The GDX-ray dataset is an important resource for the development of automated baggage screening systems, enhancing the ability of object detection models to identify prohibited items in luggage. [46]

2.4.5 OPIXray Dataset

The OPIXray dataset is aimed at improving the detection of prohibited items in baggage X-ray images. It is focused on detecting both hazardous objects and benign

`

items, providing valuable data for the development of real-time baggage scanning technologies.

2.4.5.1 Dataset Details

- Image Content: X-ray images of personal items found in hand luggage, including common electronics, liquids, and hazardous items.
- Object Categories: Includes knives, guns, liquid containers, electronics, and miscellaneous objects that are commonly found in carry-on luggage.

2.4.5.2 Key Features

- High-Resolution Images: The dataset contains high-quality X-ray images, which are important for training object detection models to recognize fine-grained features.
- Prohibited Item Detection: Specifically focuses on the detection of dangerous or prohibited items that could pose security risks.

2.4.5.3 Challenges

- Complex Object Appearance: Some objects are difficult to detect due to their complex shapes or because they are hidden within dense arrangements of personal belongings.
- Image Artifacts: X-ray images may contain noise, blur, or other artifacts that make detection more challenging.
- Object Concealment: Prohibited items are often concealed within benign objects, such as electronics or containers, making them harder to detect.

2.4.5.4 Significance:

The OPIXray dataset is crucial for advancing object detection technologies in the context of airport security, especially in detecting prohibited items concealed within hand luggage. [45]

`

CHAPTER III
METHODOLOGY

3.1 DATASET PREPARATION

The dataset used in this work is a combination of three publicly accessible datasets of X-ray images, namely SIXray, PIDXray, and CLCXray. The reason is that one of their fundamental applications in security inspection systems is their application in object detection in baggage X-ray imaging. Although these data sets contain useful content, low diversity, imbalanced, and noisy images also exist. Data augmentation and preprocessing methods were used to overcome such challenges to facilitate the training of models.

3.1.1 Data Sources

3.1.1.1 CLCXray Dataset

This dataset contains X-ray images of checked luggage intended for the identification of prohibited items, e.g. knives, scissors and liquid. Nonetheless, its limited image size and diversity require extensive preprocessing and augmentation to facilitate efficient model training. [42]

3.1.1.2 PIDXray Dataset

This dataset consists of images of personal items such as knives, guns and liquid containers and is focused on hand luggage. While realistic, the limited diversity of image data makes augmentation necessary for a robust model.[44]

3.1.1.3 SIXray Dataset

With diverse images of several setups of different luggage and contraband objects such as firearms and knives, SIXray is one of the biggest and largest X-ray databases. However, data imbalance where there are less contraband objects than regular ones and overloading image noises require appropriate preprocessing to make the most out of it. [43]

`

3.1.2 Preprocessing Techniques

In this research, no additional preprocessing techniques were manually applied to the CLCXray, PIDXray, and SIXray datasets, as the YOLO framework incorporates comprehensive built-in preprocessing functionalities. These automated processes streamline the workflow and enhance model generalization. The key preprocessing steps performed by YOLO during training include:

3.1.2.1 Image Resizing

All input images are resized to a fixed size (e.g., 640x640 pixels by default) to ensure uniformity and compatibility with the model architecture.

3.1.2.2 Normalization

Pixel values are normalized to a range between 0 and 1, facilitating faster and more stable convergence during training.

3.1.2.3 Data Augmentation

YOLO applies several augmentation techniques to improve model generalization and robustness, such as:

3.1.2.3.1 Flipping

Horizontal flipping to simulate different item orientations.

3.1.2.3.2 Scaling

Random scaling of objects within images to account for size variations.

3.1.2.3.3 Cropping

Random cropping to mimic partial occlusions and varying compositions.

3.1.2.3.4 Rotation

Minor random rotations to introduce angular variability.

3.1.2.3.5 Color Jittering

Random adjustments to brightness, contrast, and saturation.

`

3.1.2.3.6 Mosaic Augmentation

A unique technique (introduced in YOLOv4) that combines four images into one, enhancing object diversity in context.

3.1.2.4 Label Conversion

Bounding box coordinates are normalized relative to the image dimensions, ensuring consistency during training.

3.1.2.5 Anchor Matching

Used ground truth bounding box aligned to anchor boxes defined at preprocessing time so the model could learn object localization.

Subsequently by exploiting these pre-built capabilities, the model can be made to real-world fast for object detection tasks without having cumbersome additional preprocessing of custom nature -YOLO. Also, using publicly accessible dataset ensures replicability of the results as well takes full advantage of end-to-end training pipeline developed by YOLO in order to meet the goals of this research.

3.1.3 Annotation Strategies

CLCXray, PIDXray, SIXray come with boundary boxes and annotations as COCO, which is a community standard for object detection datasets.

The annotations will be of the form bounding boxes and object class labels on prohibited items in an X-ray image. The COCO format facilitates interoperability with other deep learning frameworks (e.g., PyTorch and Tensorflow), it thus makes it easier to interface models with an arbitrary object detection model.

As the annotation process is one of a few things that allow models to learn correct features of forbidden items/images and where are they in X-ray images. Thus, the annotations are consistent across these datasets gives a more effective way to evaluate model performance.

3.2 MODEL SELECTION

This research focuses on evaluating the performance of YOLOv8, YOLOv9, and YOLOv10 models for object detection in X-ray images. These models are selected for their efficiency, speed, and state-of-the-art performance, making them highly

`

suitable for real-time security inspection systems, such as baggage screening in airports.

3.2.1 YOLOv8

Representing a refinement of the YOLO framework, YOLOv8 emphasizes faster processing and improved accuracy compared to its predecessors. Its real-time detection capabilities and ability to generalize effectively across datasets make it an excellent choice for applications requiring quick responses, such as identifying prohibited items in security X-ray images.

3.2.2 YOLOv9

Building on the advancements of YOLOv8, YOLOv9 incorporates further architectural optimizations to enhance accuracy and efficiency. This version maintains the high-speed detection of earlier YOLO models while introducing improvements that address challenges like detecting smaller objects and mitigating noise, critical for X-ray security applications.

3.2.3 YOLOv10

YOLOv10 is for sophisticated tasks in which we need to detect small, overlapping and occluded objects. In this study we found its superior performance compared to other evaluated models due to the high level of clutter and noise typical in X-ray baggage scanning.

The YOLO series surpasses others in real-time objects detection models i.e., faster R-CNN or SSD — because it trades off speed for accuracy by detecting everything on the image with single shot

Furthermore, these models generally fit the particularities of challenges faced with stacking small, partially occluded prohibited items among cluttered luggage configurations in the datasets used in this research.

By selecting YOLOv8, YOLOv9 and YOLOv10 for above reasons in order to evaluate the state-of-the-art models with fine-tuned weights for real world tasks and having abilities of thousands-incoming/outgoing passengers security checkpoint.

`

3.3 TRAINING CONFIGURATIONS

The model will be trained using a 640x640 input image size for 100 epochs with a batch size of 16. It will utilize a 4070 GPU for training, and the momentum is set to 0.937 to smooth the gradient updates. Data augmentation is applied to enhance the model's robustness, and the Intersection over Union (IoU) threshold for non-max suppression is set to 0.7 during evaluation. Training will start with a pretrained model, and Automatic Mixed Precision (AMP) is enabled to speed up training and reduce memory usage by using half-precision computations.

3.4 EVALUATION METRICS

3.4.1 Classification Metrics

Classification metrics evaluate how well the model is performing in terms of object detection and classification tasks.

3.4.1.1 Accuracy

Accuracy measures how often the model's predicted class labels match the ground truth class labels.

$$Accuracy = \frac{tp + tn}{tp + fp + tn + fn} \tag{3.1}$$

where:

- TP (True Positive): Correctly predicted objects.
- FP (False Positive): Incorrectly predicted objects.
- TN (True Negative): Correctly predicted non-objects.
- FN (False Negative): Missed objects. [47]

3.4.1.2 Precision

Precision measures the proportion of positive predictions that are correct. It tells you how many of the detected objects were actually correct.

$$Precision = \frac{tp}{tp + fp} \tag{3.2}$$

`

where:

- TP (True Positive): Correctly predicted objects.
- FP (False Positive): Incorrectly predicted objects.

A low precision (<0.5) means classifier has a high number of false positive values which can be outcome imbalanced class or untuned model hyperparameters. [47]

3.4.1.3 Recall (Sensitivity)

Recall measures the proportion of actual positives (objects in the ground truth) that are correctly identified by the model.

$$Recall = \frac{tp}{tp + fn} \tag{3.3}$$

where:

- TP (True Positive): Correctly predicted objects.
- FN (False Negative): Missed objects. [47]

3.4.1.4 F1-Score

The F1-score is the harmonic mean of precision and recall. It balances the two metrics and is often used when you need a single score to assess performance.

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} \tag{3.4}$$

A high F1-score indicates a good balance between precision and recall. [47]

3.4.1.5 Intersection over Union (IoU)

IoU is a metric that measures the overlap between the predicted bounding box and the ground truth bounding box.

$$IoU = \frac{Area\ of\ overlap}{Area\ of\ union} \tag{3.5}$$

`

IoU=Area of overlap/Area of union

IoU is a critical metric in object detection tasks, as it is used to determine whether a prediction is correct (a True Positive) or a False Positive. [48]

3.4.2 Object Detection Metrics

Object detection involves both classification and localization (bounding box prediction), so it requires additional evaluation metrics:

3.4.2.1 mAP (mean Average Precision)

mAP is a widely used metric in object detection that computes the average precision across multiple classes and multiple IoU thresholds.

AP (Average Precision) is calculated by averaging precision at different recall levels, usually from 0 to 1.

$$AP = \int_0^1 P(r)dr \tag{3.6}$$

Where P(r) is the precision at recall level r. mAP averages the AP across all object categories.

$$mAP = \frac{1}{C} \sum_{c=1}^{C} AP_c \tag{3.7}$$

Where AP_c is the average precision for class c

C is the total number of classes. mAP provides a comprehensive evaluation of a detection model's ability to both classify and localize objects. [48]

3.4.2.2 mAP@50, mAP@75, and mAP@[IoU thresholds]

- **mAP@50:** This is the mean average precision calculated using an IoU threshold of 0.5. If the intersection over union of a predicted box and the ground truth box is greater than or equal to 0.5, it is considered a true positive.
- **mAP@75:** This is calculated using a stricter IoU threshold of 0.75.

`

- **mAP@[IoU thresholds]:** Some benchmarks calculate mAP at multiple IoU thresholds (e.g., [0.5:0.95] or [0.5, 0.75]). This provides a more nuanced evaluation of model performance. [48]

3.4.2.3 Average Recall (AR)

Average Recall measures how well the model recalls objects at different recall levels and is averaged across different IoU thresholds.

$$AR = \frac{1}{N} \sum_{i=1}^{N} Recall_i \tag{3.8}$$

Where N: The number of different IoU thresholds considered. $Recall_i$ the recall value at the i-th IoU threshold.

AR complements Average Precision (AP) by focusing on the model's ability to find all relevant objects (high recall). It provides a holistic view of recall performance across a range of IoU thresholds rather than relying on a single threshold like 0.5. [48]

3.4.3 Training Metrics Tracking

During training, various metrics are tracked to monitor the model's progress and detect overfitting or underfitting. These include:

3.4.3.1 Localization Loss (Bounding Box Regression)

Measures the error between the predicted bounding boxes and ground truth using metrics like CIoU:

$$L_{loc} = 1 - IoU(b_p, b_t) + \frac{\rho^2(b_p, b_t)}{c^2} + \alpha v \tag{3.9}$$

Where:

- IoU: Intersection over Union.
- ρ^2: Squared distance between box centers.
- c: Diagonal length of the smallest enclosing box.
- v: Aspect ratio consistency. [49]

`

3.4.3.2 Confidence Loss

Measures the error in the predicted objectness score:

$$L_{conf} = \sum_{i=1}^{S} \sum_{j=1}^{B} BCE(P_{ij}, P_{i^*j}) \tag{3.10}$$

Where:

- BCE: Binary Cross-Entropy loss.
- P_{ij}: Predicted confidence score.
- P_{i^*j}: Ground truth confidence score. [49]

3.4.3.3 Classification Loss

Measures the error in predicted class probabilities:

$$L_{cls} = -\sum_{k=1}^{C} y_k \log \hat{y}_k \tag{3.11}$$

Where:

- y_k: Ground truth label for class k.
- $\hat{y}_k$: Predicted probability for class k. [49]

3.4.3.4 Total Loss

$$\mathcal{L}_{total} = \lambda_{loc}\mathcal{L}_{loc} + \lambda_{conf}\mathcal{L}_{conf} + \lambda_{cls}\mathcal{L}_{cls} \tag{3.12}$$

3.4.3.5 Learning Curves

- **Accuracy Curve:** Tracks classification accuracy over the course of training.
- **Loss Curve:** Tracks the total training loss to identify whether the model is improving and if it has converged.
- **Precision-Recall Curve:** Plots precision against recall at various thresholds and helps visualize the tradeoff between false positives and false negatives.

`

- **ROC (Receiver Operating Characteristic) Curve:** A plot of the true positive rate (recall) against the false positive rate, showing the trade-off between sensitivity and specificity.
- **AUC (Area Under Curve):** A value representing the overall ability of the classifier to distinguish between positive and negative classes.
- **Confusion Matrix:** A confusion matrix provides a detailed breakdown of true positives, false positives, true negatives, and false negatives, helping identify where the model is making mistakes. [50]

`

CHAPTER IV
EXPERIMENTAL RESULTS

4.1 CLCXRAY

The evaluation of YOLO models (Nano, Small, Medium) using the CLCXray dataset revealed significant performance differences across metrics and categories. YOLOv10 Medium consistently achieved the highest true positive (TP) counts across categories, including a high count for Plastic Bottle, outperforming YOLOv8 and YOLOv9. Based on the graphs, YOLOv10 Medium had approximately 444 TPs for Plastic Bottle. Small models, such as YOLOv10 Small, struck a balance between performance and computational efficiency, achieving approximately 433 TPs for Plastic Bottle. Nano models, designed for lightweight applications, detected slightly fewer true positives, with YOLOv10 Nano achieving approximately 433 TPs for Plastic Bottle.

In terms of false positives (FP), YOLOv10 Medium generally had the lowest rates across categories, including approximately 42 FPs for Plastic Bottle. Small models had slightly higher FPs, with YOLOv10 Small showing approximately 48 FPs for Plastic Bottle. Nano models showed the highest FPs due to their simplified architecture, with YOLOv10 Nano having approximately 79 FPs for Plastic Bottle.

Qualitative analysis (based on the trends observed in the graphs and shown in Figure 13 with example detection results) suggests that YOLOv10 Medium's ability to detect objects while minimizing false positives is superior. Small models were effective but showed slightly more false positives. Nano models excelled in inference speed (not directly shown in the graphs but a general characteristic) but had the highest false positive rates. Compared to YOLOv8 and YOLOv9, YOLOv10 consistently outperformed in all configurations, achieving higher true positives and reducing false positives across most categories.

`

Figure 13: CLCXray Detection Example

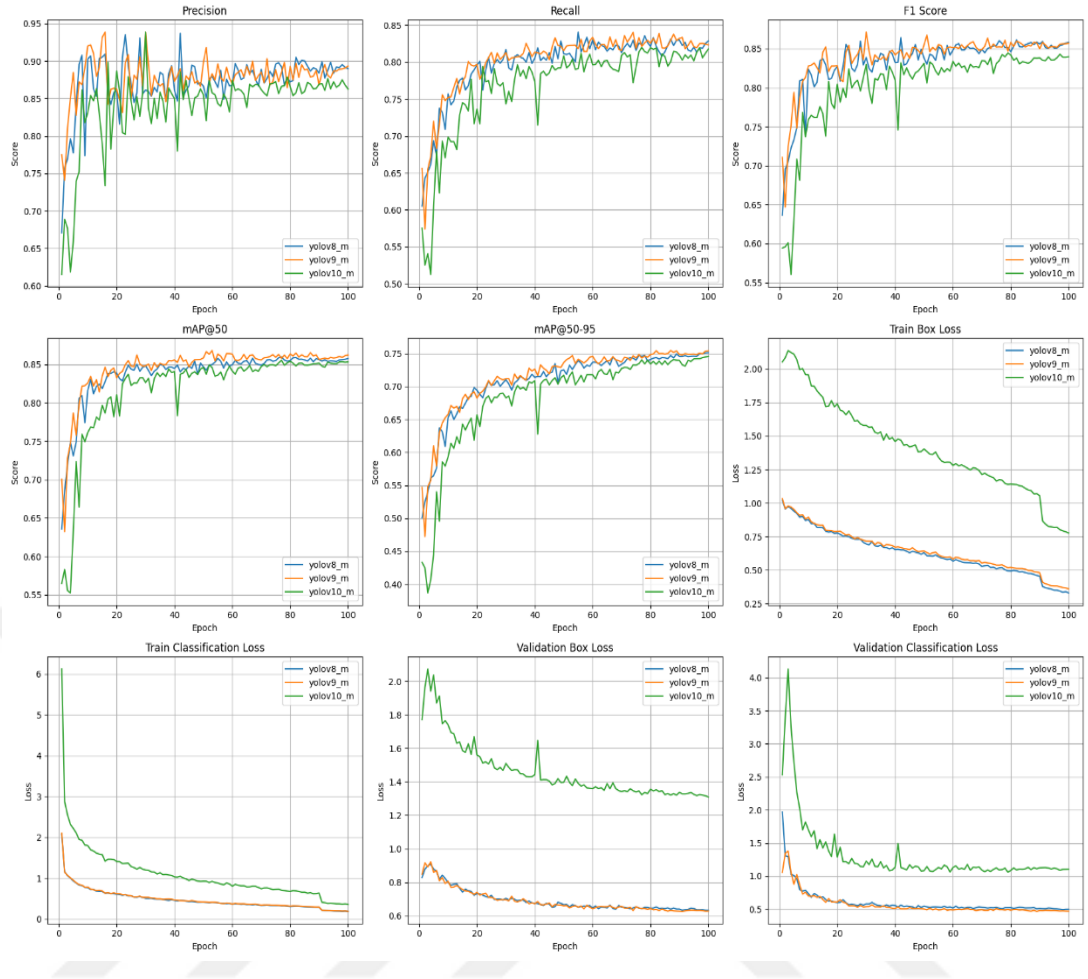
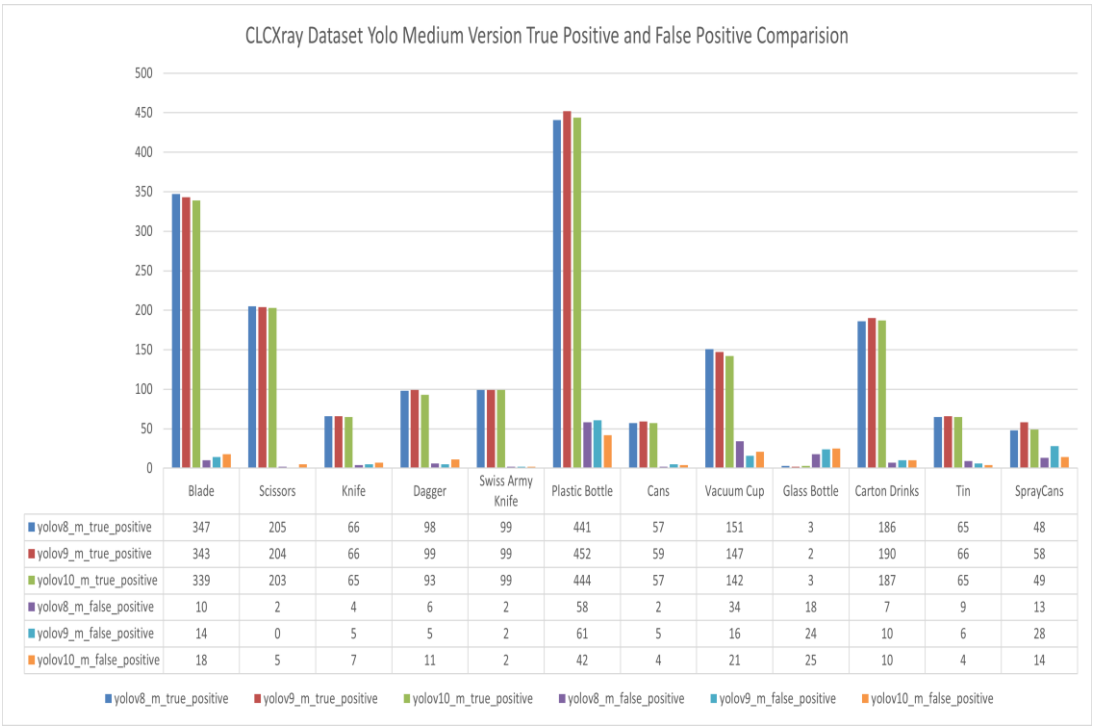


Figure 14: CLCXRray Medium Models Performance Comparison Graphs



	Blade	Scissors	Knife	Dagger	Swiss Army Knife	Plastic Bottle	Cans	Vacuum Cup	Glass Bottle	Carton Drinks	Tin	SprayCans
yolov8_m_true_positive	347	205	66	98	99	441	57	151	3	186	65	48
yolov9_m_true_positive	343	204	66	99	99	452	59	147	2	190	66	58
yolov10_m_true_positive	339	203	65	93	99	444	57	142	3	187	65	49
yolov8_m_false_positive	10	2	4	6	2	58	2	34	18	7	9	13
yolov9_m_false_positive	14	0	5	5	2	61	5	16	24	10	6	28
yolov10_m_false_positive	18	5	7	11	2	42	4	21	25	10	4	14

Figure 15: CLCXRray Medium Models Performance TP/FP Graphs

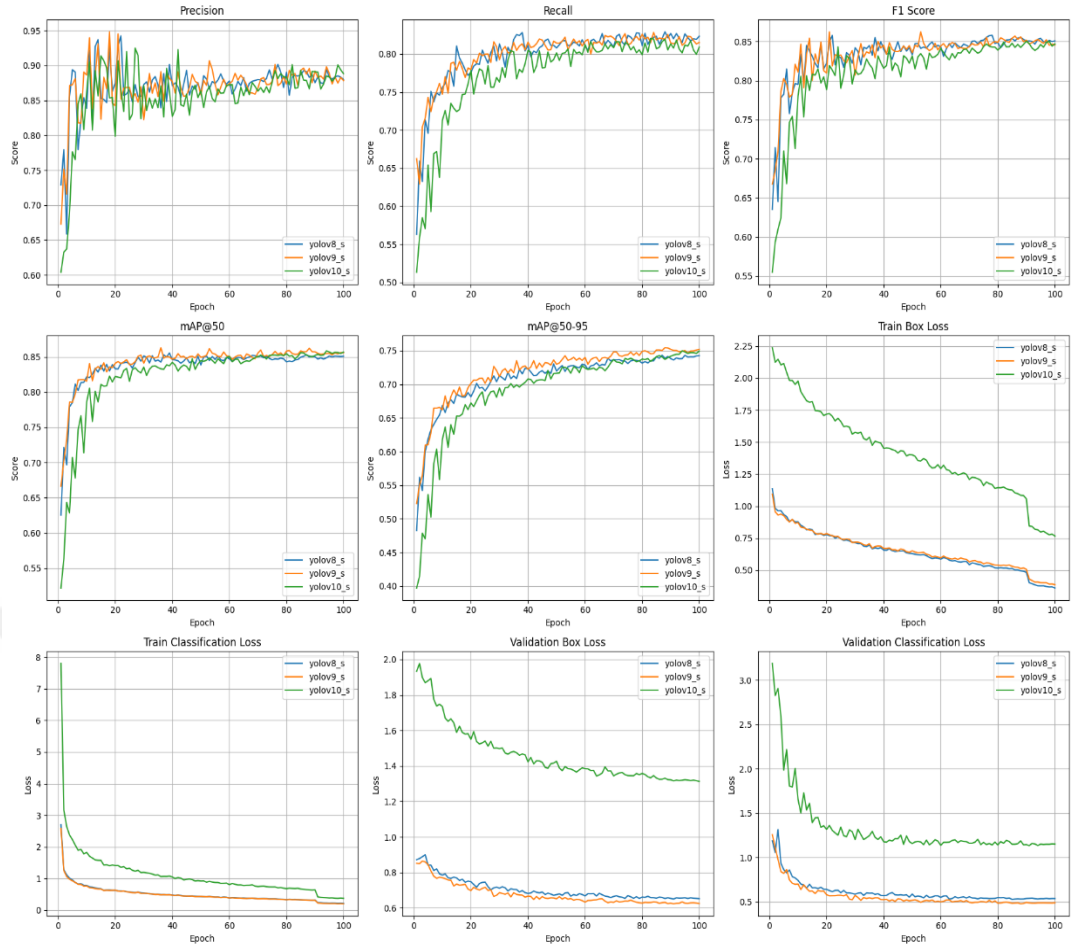
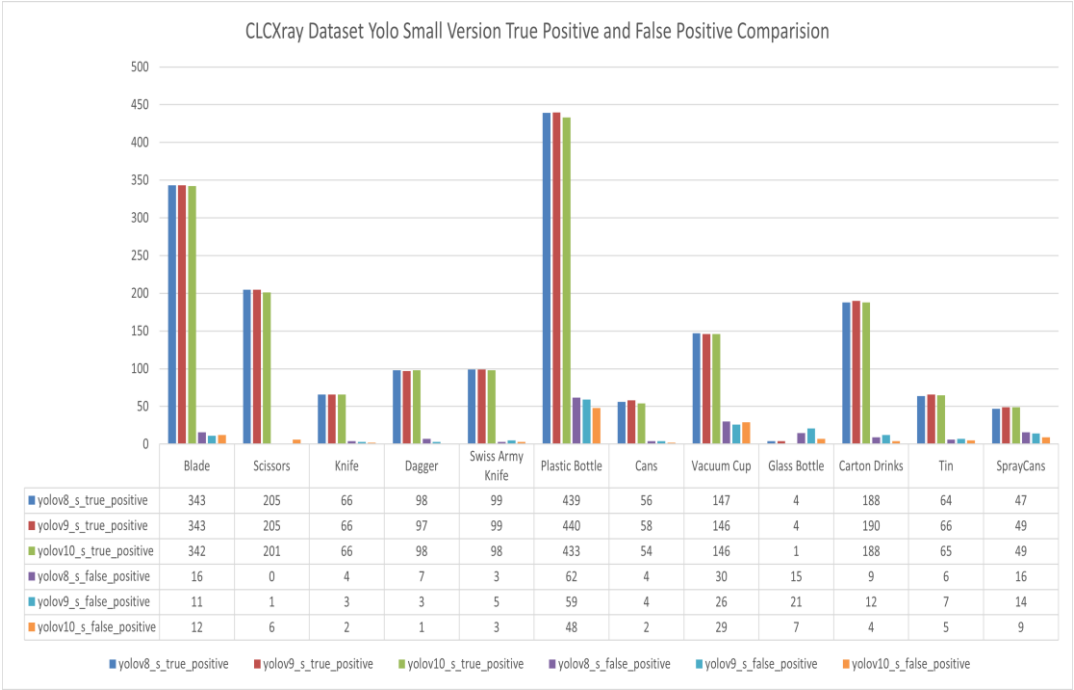


Figure 16: CLCXRray Small Models Performance Comparison Graphs



	Blade	Scissors	Knife	Dagger	Swiss Army Knife	Plastic Bottle	Cans	Vacuum Cup	Glass Bottle	Carton Drinks	Tin	SprayCans
yolov8_s_true_positive	343	205	66	98	99	439	56	147	4	188	64	47
yolov9_s_true_positive	343	205	66	97	99	440	58	146	4	190	66	49
yolov10_s_true_positive	342	201	66	98	98	433	54	146	1	188	65	49
yolov8_s_false_positive	16	0	4	7	3	62	4	30	15	9	6	16
yolov9_s_false_positive	11	1	3	3	5	59	4	26	21	12	7	14
yolov10_s_false_positive	12	6	2	1	3	48	2	29	7	4	5	9

Figure 17: CLCXRray Small Models Performance TP/FP Graphs

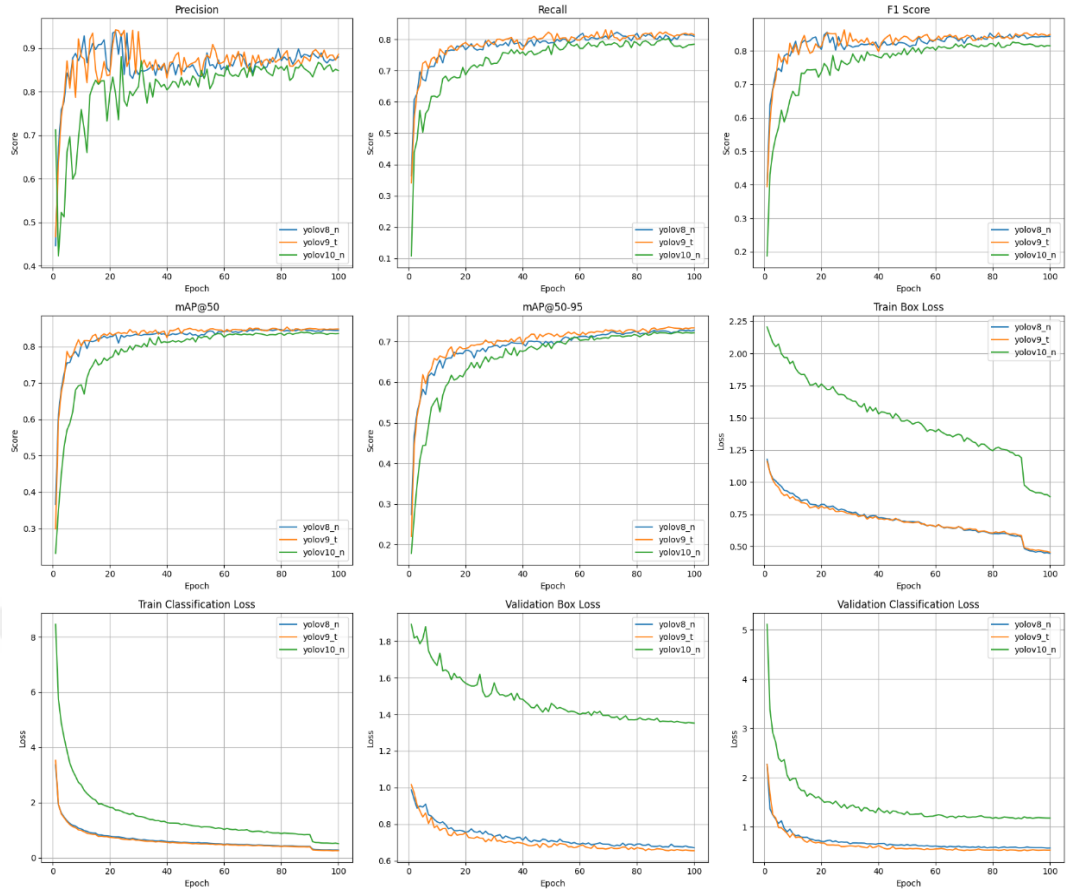
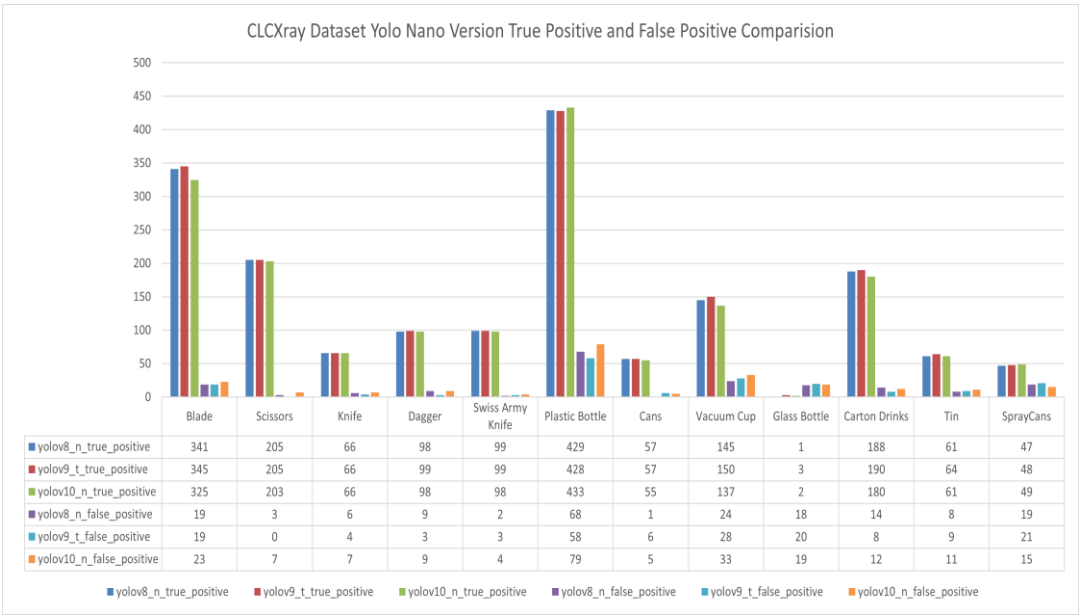


Figure 18: CLCXRray Nano Models Performance Comparison Graphs



	Blade	Scissors	Knife	Dagger	Swiss Army Knife	Plastic Bottle	Cans	Vacuum Cup	Glass Bottle	Carton Drinks	Tin	SprayCans
yolov8_n_true_positive	341	205	66	98	99	429	57	145	1	188	61	47
yolov9_t_true_positive	345	205	66	99	99	428	57	150	3	190	64	48
yolov10_n_true_positive	325	203	66	98	98	433	55	137	2	180	61	49
yolov8_n_false_positive	19	3	6	9	2	68	1	24	18	14	8	19
yolov9_t_false_positive	19	0	4	3	3	58	6	28	20	8	9	21
yolov10_n_false_positive	23	7	7	9	4	79	5	33	19	12	11	15

Figure 19: CLCXRray Nano Models Performance TP/FP Graphs

`

4.2 PIDXRAY

The evaluation of YOLO models (Nano, Small, Medium) using the PIDXray dataset revealed significant performance differences across metrics and categories. YOLOv10 Medium consistently achieved the highest true positive (TP) counts across complex categories like Powerbank (1,010 TPs) and Knife (649 TPs), outperforming YOLOv8 and YOLOv9. Small models, such as YOLOv10 Small, struck a balance between performance and computational efficiency, with 595 TPs for Knife. Nano models, designed for lightweight applications, detected fewer true positives, such as 520 TPs for Knife. In terms of false positives (FP), YOLOv10 Medium had the lowest rates (Powerbank: 84 FPs, Knife: 40 FPs), followed by Small models (Powerbank: 115 FPs, Knife: 48 FPs). Nano models showed higher FPs due to their simplified architecture (Powerbank: 140 FPs, Knife: 55 FPs).

Qualitative analysis highlighted YOLOv10 Medium's ability to detect complex and overlapping objects while minimizing false positives, as shown in Figure 20 with example detection results. Small models were effective in simpler scenarios but struggled with object overlap and texture complexity. Nano models excelled in inference speed but often misclassified objects with intricate textures, such as confusing a Hammer with a Wrench. Compared to YOLOv8 and YOLOv9, YOLOv10 consistently outperformed in all configurations, achieving higher true positives and reducing false positives. For example, YOLOv10 Medium detected 972 TPs for Guns compared to 883 TPs (Small) and 810 TPs (Nano). Ablation studies emphasized the tradeoffs: Medium models offered the best accuracy, Small models provided efficiency for mid-tier devices, and Nano models prioritized speed but sacrificed accuracy, particularly for complex features or crowded scenes. YOLOv10 demonstrated superior performance across all metrics, making it a reliable choice for various scenarios.

Figure 20: PIDXray Detection Results Examples

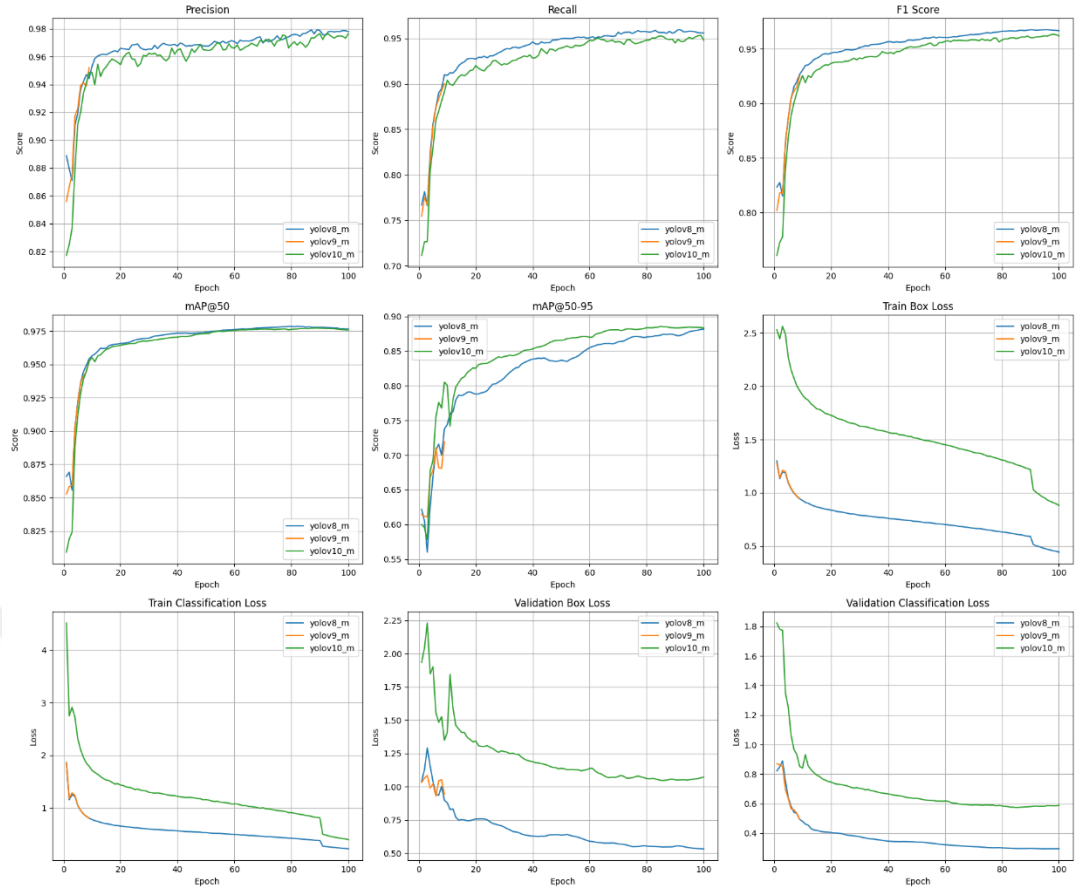
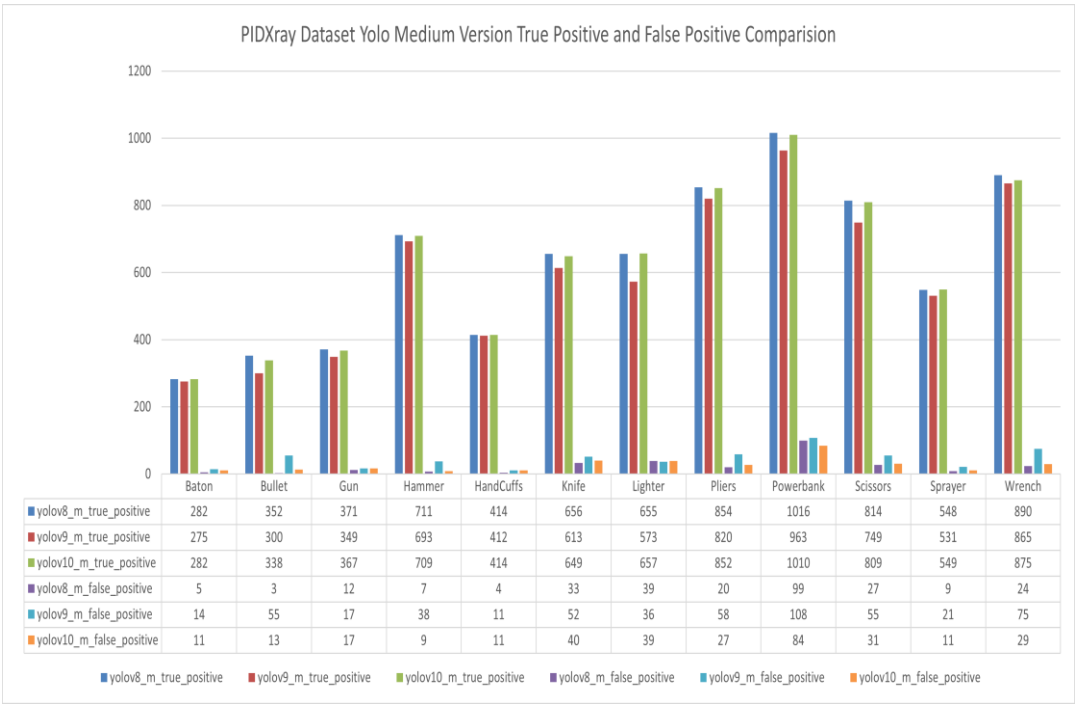


Figure 21: PIDXray Medium Models Performance Comparison Graphs



	Baton	Bullet	Gun	Hammer	HandCuffs	Knife	Lighter	Pliers	Powerbank	Scissors	Sprayer	Wrench
yolov8_m_true_positive	282	352	371	711	414	656	655	854	1016	814	548	890
yolov9_m_true_positive	275	300	349	693	412	613	573	820	963	749	531	865
yolov10_m_true_positive	282	338	367	709	414	649	657	852	1010	809	549	875
yolov8_m_false_positive	5	3	12	7	4	33	39	20	99	27	9	24
yolov9_m_false_positive	14	55	17	38	11	52	36	58	108	55	21	75
yolov10_m_false_positive	11	13	17	9	11	40	39	27	84	31	11	29

Figure 22: PIDXray Medium Models Performance TP/FP Graphs

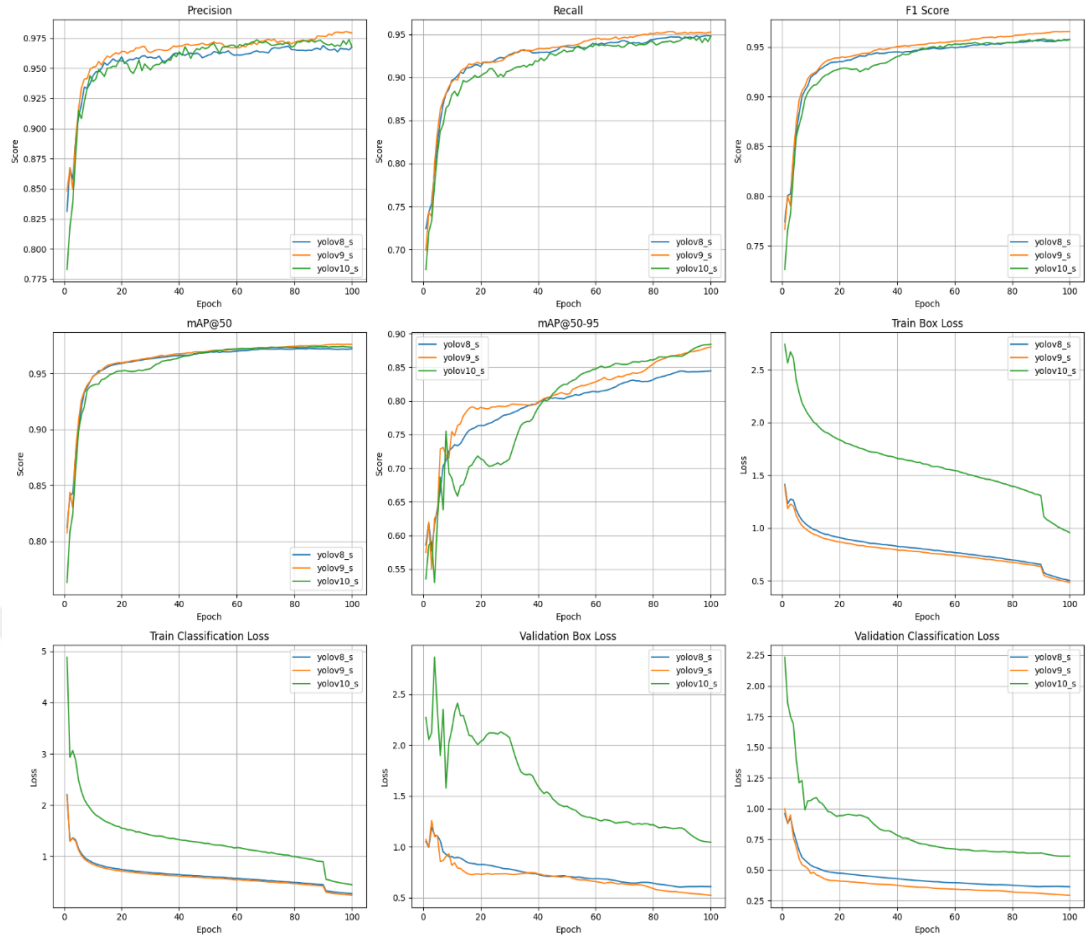
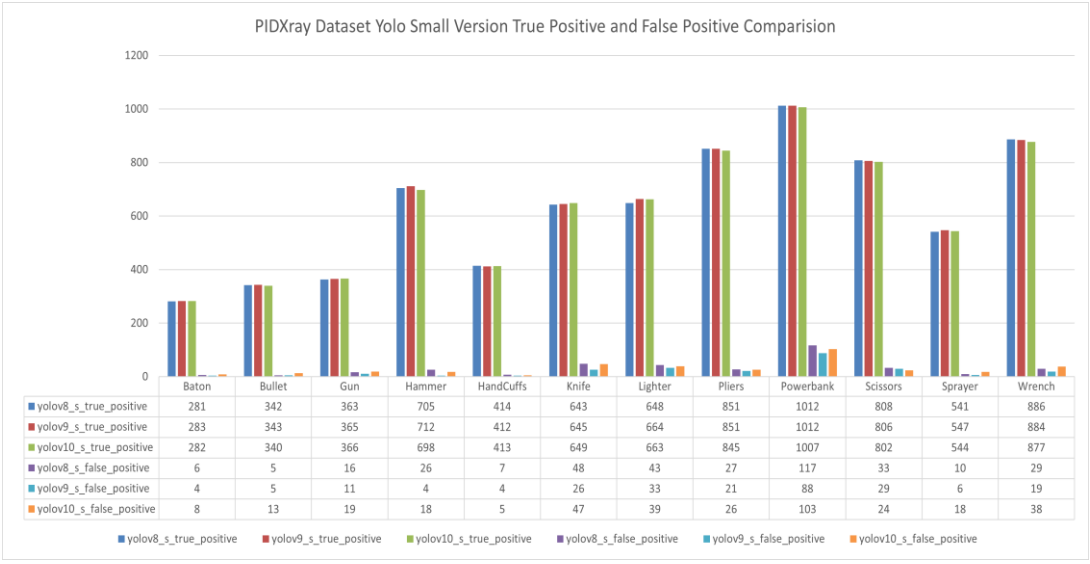


Figure 23: PIDXray Small Models Performance Comparison Graphs



	Baton	Bullet	Gun	Hammer	HandCuffs	Knife	Lighter	Pliers	Powerbank	Scissors	Sprayer	Wrench
yolov8_s_true_positive	281	342	363	705	414	643	648	851	1012	808	541	886
yolov9_s_true_positive	283	343	365	712	412	645	664	851	1012	806	547	884
yolov10_s_true_positive	282	340	366	698	413	649	663	845	1007	802	544	877
yolov8_s_false_positive	6	5	16	26	7	48	43	27	117	33	10	29
yolov9_s_false_positive	4	5	11	4	4	26	33	21	88	29	6	19
yolov10_s_false_positive	8	13	19	18	5	47	39	26	103	24	18	38

Figure 24: PIDXray Small Models Performance TP/FP Graphs

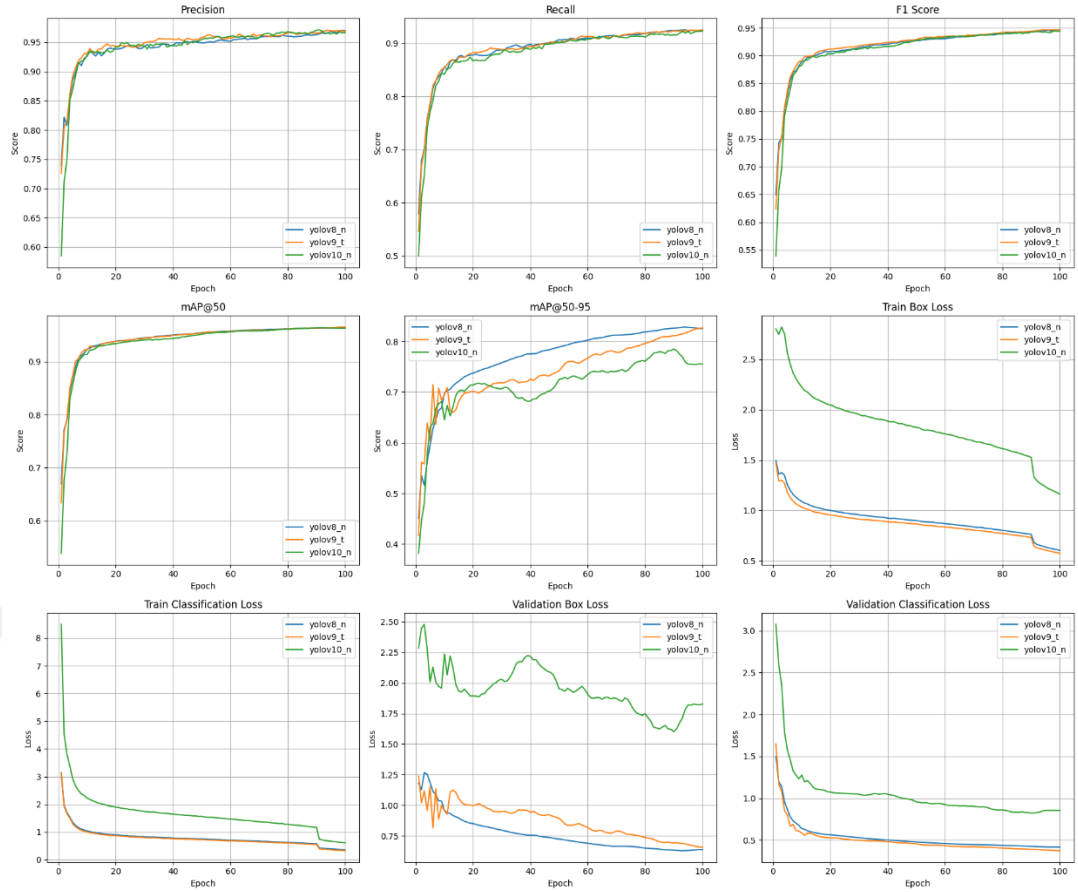
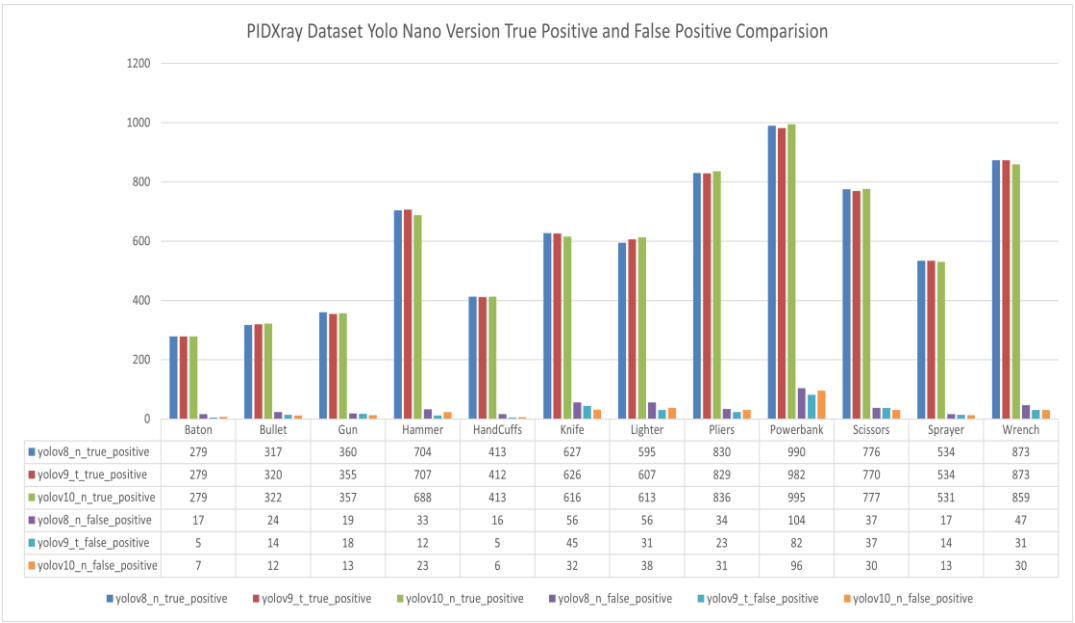


Figure 25: PIDXray Nano Models Performance Comparison Graphs



	Baton	Bullet	Gun	Hammer	HandCuffs	Knife	Lighter	Pliers	Powerbank	Scissors	Sprayer	Wrench
yolov8_n_true_positive	279	317	360	704	413	627	595	830	990	776	534	873
yolov9_t_true_positive	279	320	355	707	412	626	607	829	982	770	534	873
yolov10_n_true_positive	279	322	357	688	413	616	613	836	995	777	531	859
yolov8_n_false_positive	17	24	19	33	16	56	56	34	104	37	17	47
yolov9_t_false_positive	5	14	18	12	5	45	31	23	82	37	14	31
yolov10_n_false_positive	7	12	13	23	6	32	38	31	96	30	13	30

Figure 26: PIDXray Nano Models Performance TP/FP Graphs

`

4.3 SIXRAY

The evaluation of YOLO models (Nano, Small, Medium) using the SIXray dataset revealed significant performance differences across metrics and categories. YOLOv10 Medium consistently achieved high true positive (TP) counts across categories like Pliers (over 950 TPs), outperforming YOLOv8 and YOLOv9 in minimizing False Positives. Small models, such as YOLOv10 Small, struck a balance between performance and computational efficiency, with performance close to Medium in categories like Pliers (over 950 TPs). Nano models, designed for lightweight applications, detected slightly fewer true positives in categories like Pliers (over 900 TPs). In terms of false positives (FP), YOLOv10 Medium generally had the lowest rates across categories. Small models had slightly higher FPs, and Nano models showed higher FPs due to their simplified architecture.

Qualitative analysis (based on the trends observed in the graphs) suggests YOLOv10 Medium's ability to minimize false positives as shown in Figure 27. Small models were effective but showed slightly more false positives. Nano models excelled in inference speed but had the highest false positive rates. Compared to YOLOv8 and YOLOv9, YOLOv10 consistently performed well in minimizing false positives across most categories. For example, focusing on minimizing False Positives YOLOv10 Medium had good results in Gun, Wrench categories. Ablation studies (inferred from the performance differences across model sizes) emphasized the tradeoffs: Medium models offered the best accuracy, Small models provided efficiency for mid-tier devices, and Nano models prioritized speed but sacrificed accuracy, particularly evident in the higher false positive rates. YOLOv10 demonstrated strong performance in minimizing false positives across all metrics and model sizes on the SIXray dataset, making it a reliable choice for various scenarios.

Figure 27: SIXRay Detection Results Examples

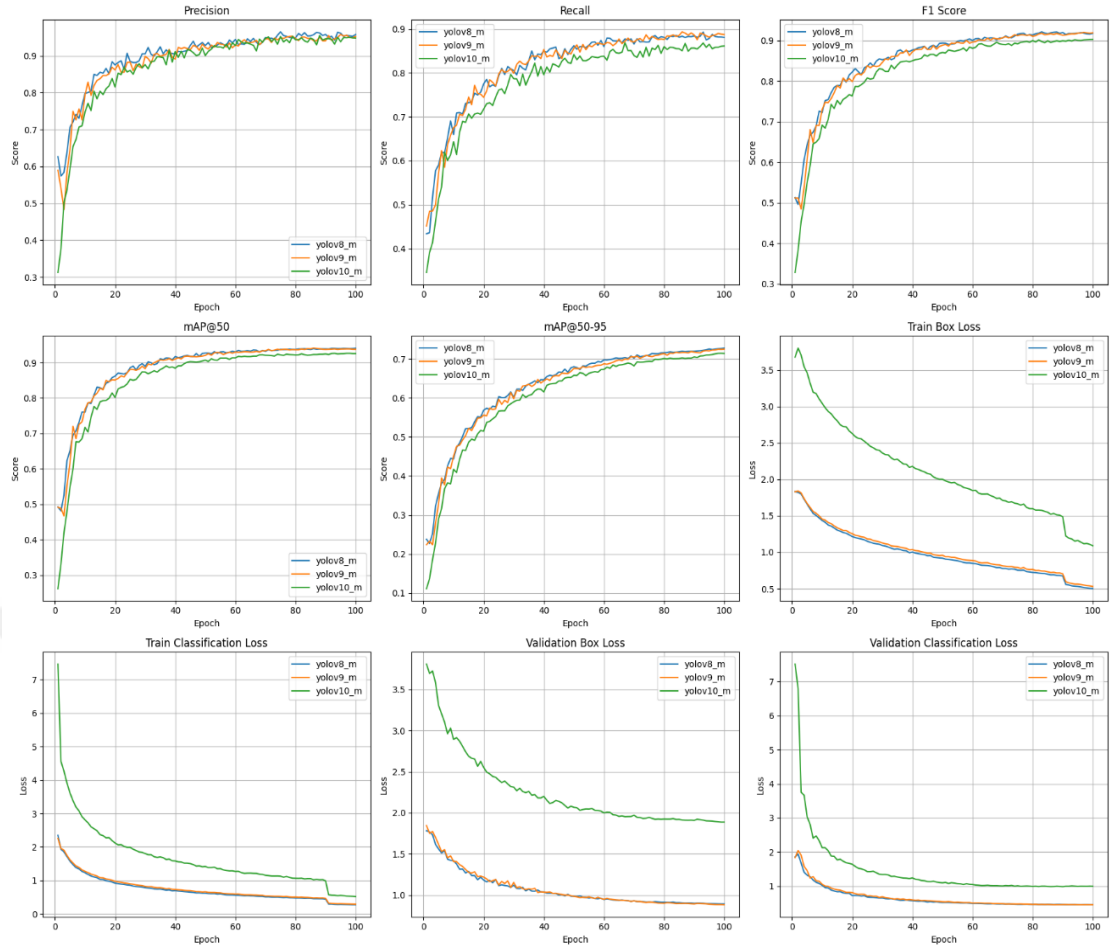
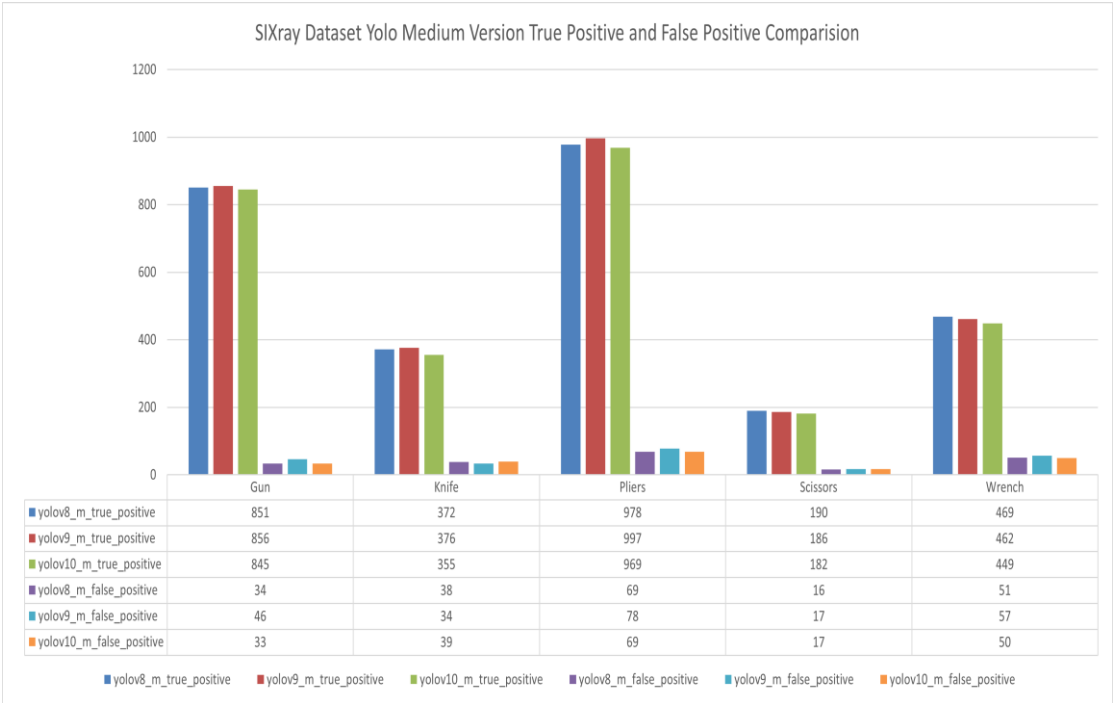


Figure 28: SIXray Medium Models Performance Comparison Graphs



	Gun	Knife	Pliers	Scissors	Wrench
yolov8_m_true_positive	851	372	978	190	469
yolov9_m_true_positive	856	376	997	186	462
yolov10_m_true_positive	845	355	969	182	449
yolov8_m_false_positive	34	38	69	16	51
yolov9_m_false_positive	46	34	78	17	57
yolov10_m_false_positive	33	39	69	17	50

Figure 29: SIXray Medium Models Performance TP/FP Graphs

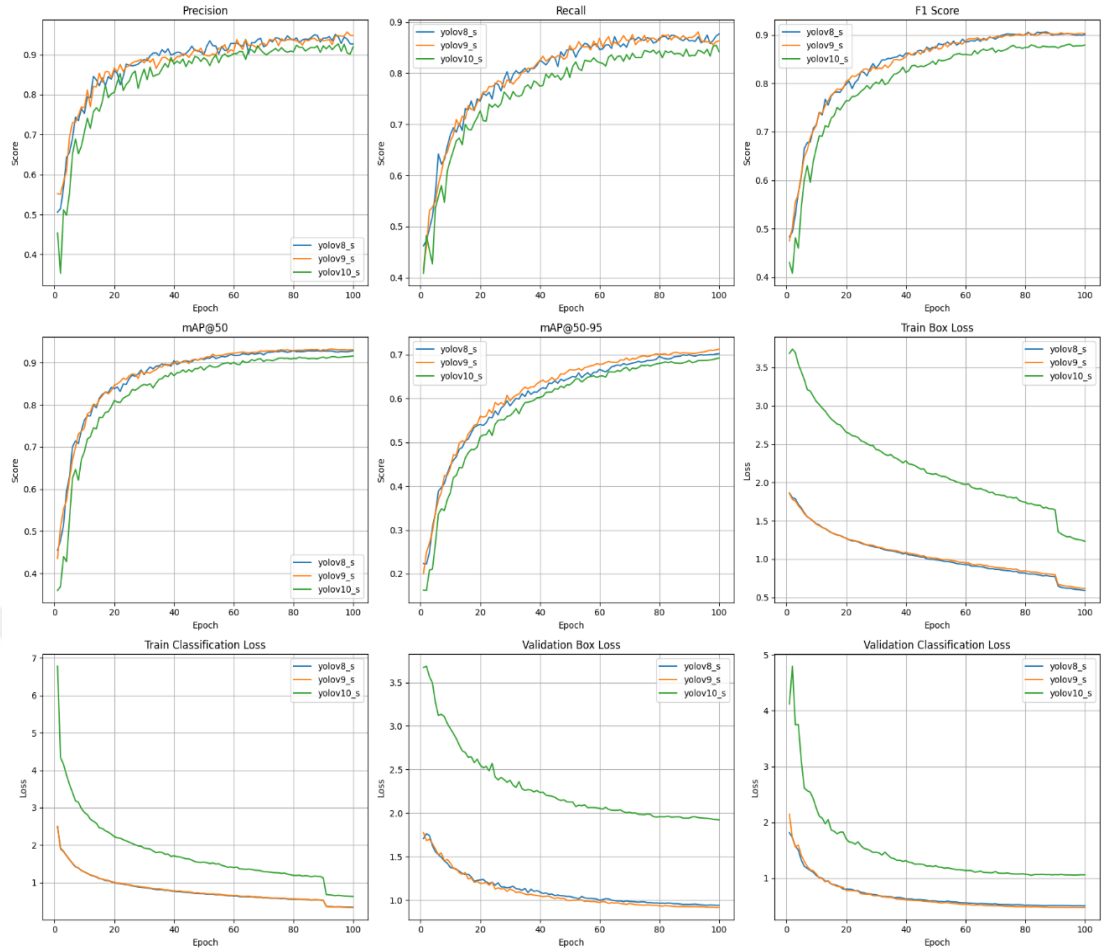
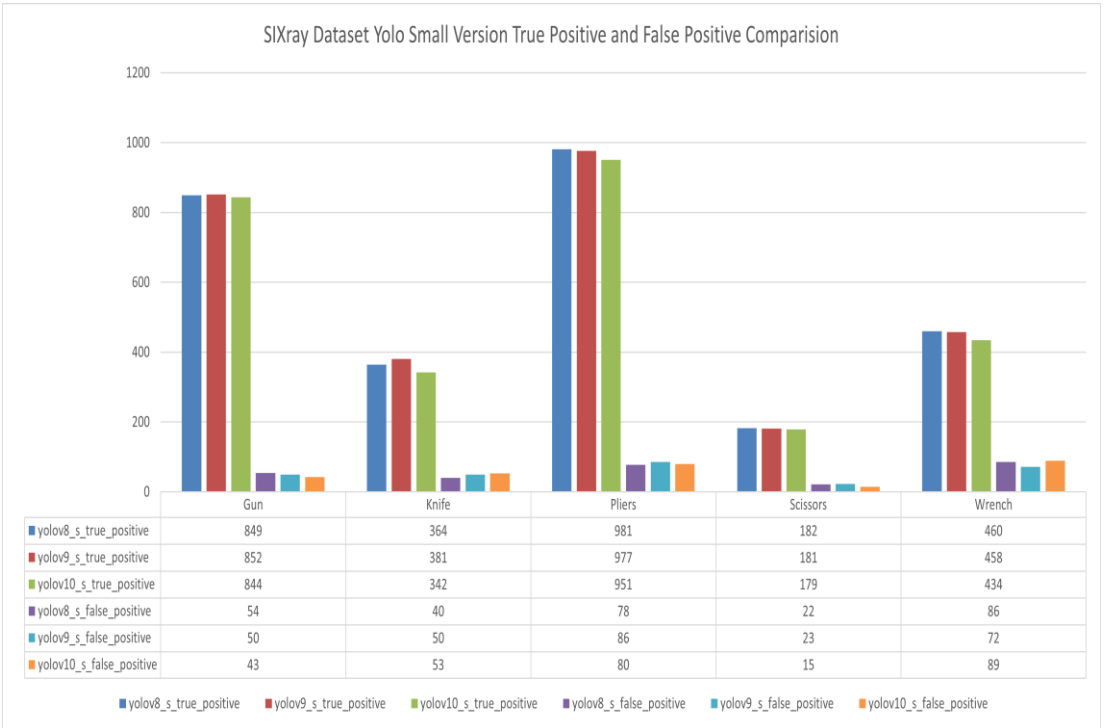


Figure 30: SIXray Small Models Performance Comparison Graphs



	Gun	Knife	Pliers	Scissors	Wrench
yolov8_s_true_positive	849	364	981	182	460
yolov9_s_true_positive	852	381	977	181	458
yolov10_s_true_positive	844	342	951	179	434
yolov8_s_false_positive	54	40	78	22	86
yolov9_s_false_positive	50	50	86	23	72
yolov10_s_false_positive	43	53	80	15	89

Figure 31: SIXray Small Models Performance TP/FP Graphs

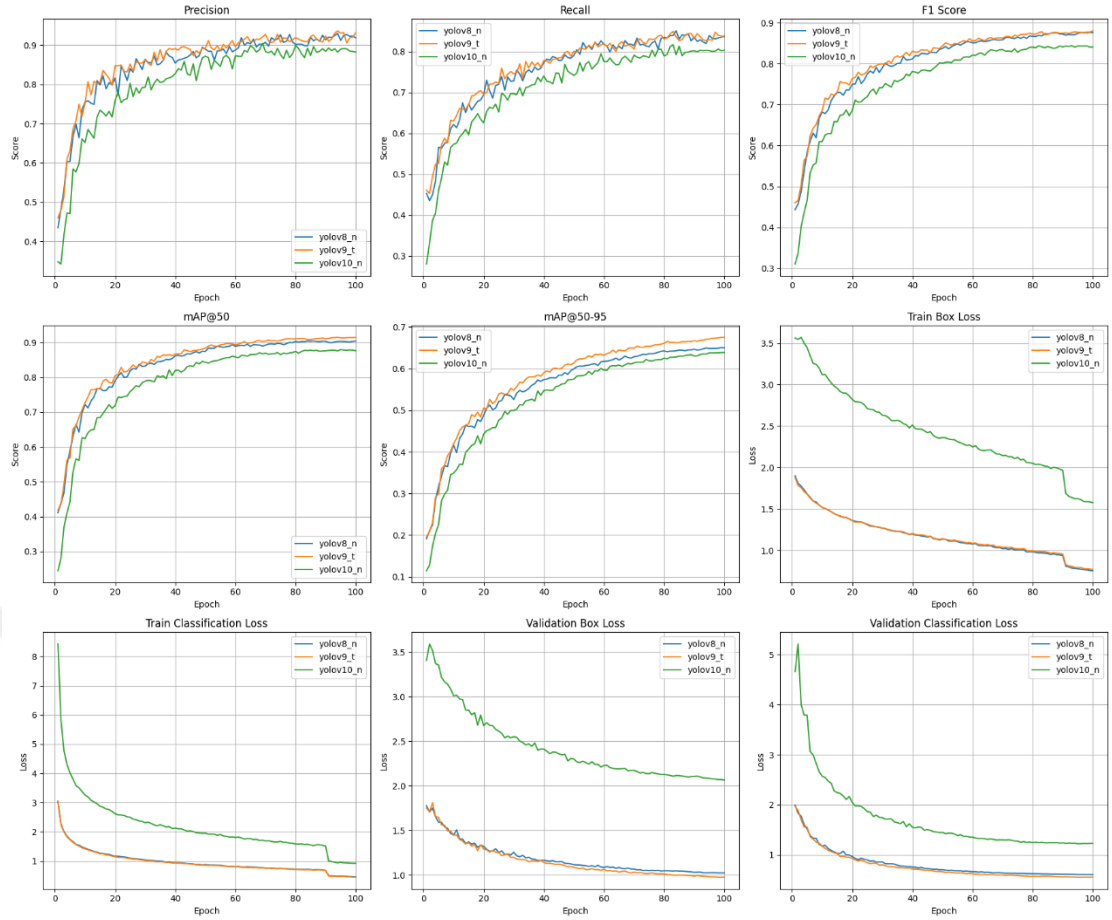
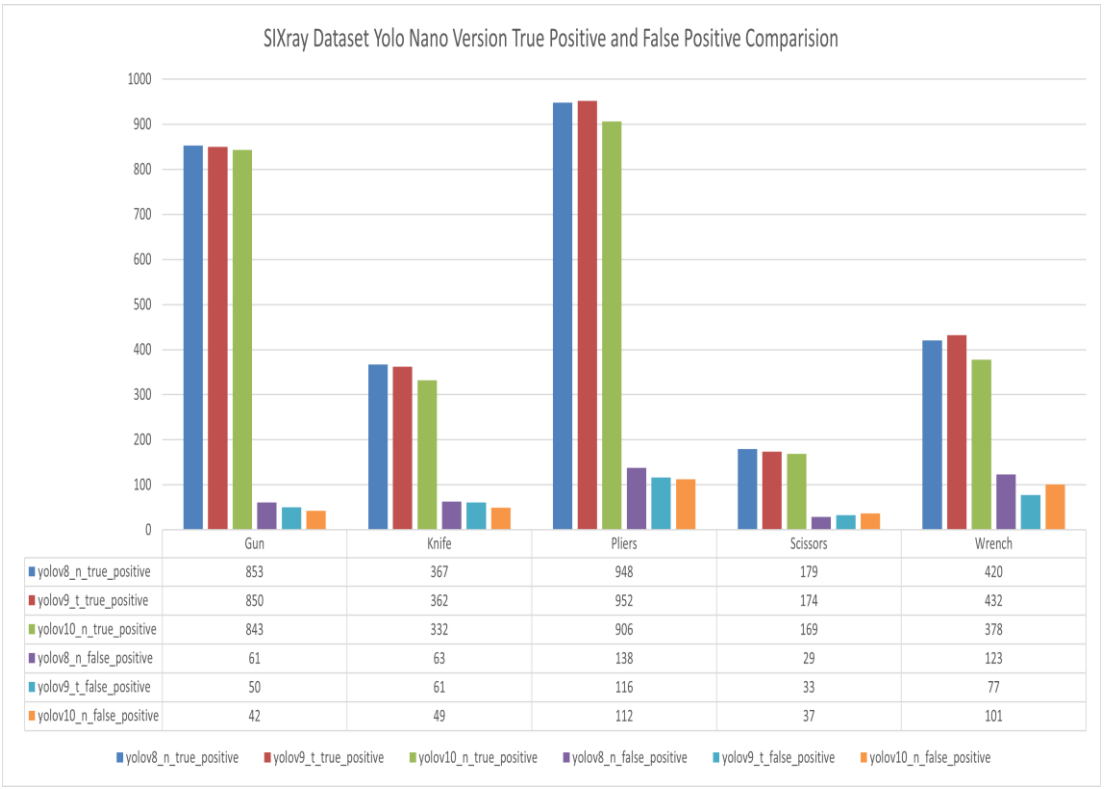


Figure 32: SIXray Nano Models Performance Comparison Graphs



	Gun	Knife	Pliers	Scissors	Wrench
yolov8_n_true_positive	853	367	948	179	420
yolov9_t_true_positive	850	362	952	174	432
yolov10_n_true_positive	843	332	906	169	378
yolov8_n_false_positive	61	63	138	29	123
yolov9_t_false_positive	50	61	116	33	77
yolov10_n_false_positive	42	49	112	37	101

Figure 33: SIXray Nano Models Performance TP/FP Graphs

`

CHAPTER V
CONCLUSION AND DISCUSSION

5.1 KEY INSIGHTS

This research thoroughly evaluated the performance of the latest YOLO models—YOLOv8, YOLOv9, and YOLOv10—on three prominent X-ray baggage screening datasets: CLCXray, PIDXray, and SIXray. The findings demonstrate significant advancements in object detection for X-ray imagery, particularly addressing challenges such as occluded and overlapping objects, high variability in object shapes, sizes, and orientations, as well as low resolution and noisy imaging conditions.

YOLOv10 was the best model, and it had the highest detection rate and inference speed with low computation requirements, adequately overcoming such challenges. This balance provides it with a very appropriate for application in high-throughput settings such as airports. YOLOv8 and YOLOv9 also performed competitively, with strengths in specific scenarios, such as detecting objects in less cluttered settings or under constrained computational resources. These results validate the potential of the YOLO framework in operational security systems as real-time object detection of effective solution.

5.2 LIMITATIONS

- **Dataset Constraints:** The size and diversity of the datasets used, while significant, may not fully represent the variety of objects and configurations encountered in real-world scenarios.
- **Computational Resources:** The experiments relied on fixed computational resources, which limited the ability to test the models' scalability in more demanding environments.
- **Model Generalization:** Although the models operated on the selected datasets, generalizability to out-of-sample data or even even more distinct imaging modalities hasn't been tested. Elimination of these

`

deficits would better indicate what is possible in the way of model capability.

5.3 APPLICATIONS

The outcomes of these research have implications towards improvement of automatic baggage inspection. Through greater performance in detection and higher-speed operations, these YOLO architectures would provide more secure forbidden object detection by security agents to counter human errors and general safety improvement. Other than that, faster inference performance of YOLOv10 makes it ideal for real-time performance, thus ideal for high-throughput environments like checkpoints in airports. Integration would create more seamless operations with reduced bottlenecks and general improvement in user experience.

5.4 FUTURE WORK

Following up on the findings of this research, some productive avenues for future research may be:

- **Model Enhancement:** Design and performance comparison of next generation of model YOLOv11. Follow-up research would determine how exactly object detection performance, inference performance, and computational cost have been improved in YOLOv11 compared to its predecessors. There would have to be special emphasis on architecture for object overlap cases as well as low-contrast images that typify X-ray screening.
- **Dataset Expansion:** Expansion of larger diversified databases of more objects, more orientations of imaging, and more realistic conditions.
- **Multi-modal Methods:** Investigation of multimodal combination of X-ray imaging with other imaging methods such as 3D or infrared for more robust detection.
- **Real-time Applications:** Focusing on optimizing models for real-time detection in dynamic and resource-constrained environments, enabling broader deployment.
- **Explainability:** Incorporating explainable AI techniques to make model predictions more interpretable for end-users.

`

5.5 CONCLUSION

This thesis was on object detection in topical X-ray baggage inspection systems. Through leveraging on the model of YOLO, research was achieved that significantly enhanced object detection performance, performance in inferences, as well as computational performance with that of YOLOv10 having performance at best in general performance. All these results reflect on potential in practice of next-generation object detection technology that is founded on deep-learning to solve security concerns in delivering more effective as well as more efficient automatic inspection in future. All these results offer foundation for future developments in object detection technology as well as for object detection technology in deployment in operating environment scenarios.

`

REFERENCES

[1] CONWAY Alison, KAMGA Camille, YAZICI Anil and SINGHAL Abhishek (2012), “Challenges in Managing Centralized Taxi Dispatching at High-Volume Airports: Case Study of John F. Kennedy International Airport, New York City”, *Transportation Research Record*, Vol. 2300 No. 1, pp. 83-90.

[2] OU Xiangyu, CHEN Xue, XU Xianning, XIE Lili, CHEN Xiaofeng, HONG Zhongzhu, BAI Hua, LIU Xiaowang, CHEN Qiushui, LI Lin and YANG Huanghao (2021), "Recent development in X-ray imaging technology: Future and challenges", *AAAS Research*, Vol. 2021 No. 5 pp. 1-18.

[3] HE Kaiming, ZHANG Xiangyu, REN Shaoqing and SUN Jian (2016), “Deep residual learning for image recognition”, *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770-778, Las Vegas.

[4] GIRSHICK Ross B., DONAHUE Jeff, DARRELL Trevor and MALIK Jitendra (2014), “Rich feature hierarchies for accurate object detection and semantic segmentation”, *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 580-587, Colombus.

[5] LIU Wei, ANGUELOV Dragomir, ERHAN Dumitru, SZEGEDY Christian, REED Scott, FU Cheng-Yang and BERG Alexander C. (2016), “SSD: Single shot multibox detector”, *In, Computer Vision – ECCV 2016*, Eds. Bastian Leibe, Jiri Matas, Nicu Sebe and Max Welling, pp. 21-37, Springer, Amsterdam.

[6] REDMON Joseph, DIVVALA Santosh, GIRSHICK Ross and FARHADI Ali (2016), “You only look once: Unified, real-time object detection”, *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 779-788, Las Vegas.

[7] REDMON Joseph and FARHADI Ali (2017), “YOLO9000: Better, faster, stronger”, *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6517-6525, Honolulu.

`

[8] REDMON Joseph and FARHADI Ali (2018), “YOLOv3: An incremental improvement”, DOI: 10.48550/arXiv.1804.02767.

[9] BOCHKOVSKIY Alexey, WANG Chien-Yao and LIAO Hong-Yuan Mark (2020), “YOLOv4: Optimal speed and accuracy of object detection”, DOI: 10.48550/arXiv.2004.10934.

[10] LOWE David G. (2004), “Distinctive Image Features from Scale-Invariant Keypoints”, *International Journal of Computer Vision*, Vol. 60, pp. 91–110,

[11] BAY Herbert, TUYTELAARS Tinne and VAN GOOL Luc (2006), “SURF: Speeded Up Robust Features”, *In, Computer Vision – ECCV 2006*, Eds. Aleš Leonardis, Horst Bischof and Axel Pinz, pp. 404–417, Springer, Graz.

[12] DALAL Navneet and TRIGGS Bill (2005), “Histograms of Oriented Gradients for Human Detection”, *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, pp. 886–893, San Diego.

[13] LECUN Yann, BOSER Bernhard E., DENKER John S., HENDERSON Donnie, HOWARD Richard E., JACKEL Lawrence D. and SINGER Y. (1989), “Backpropagation applied to handwritten zip code recognition”, *Neural Computation*, Vol. 1, p.541–551.

[14] GIRSHICK Ross (2015), “Fast R-CNN”, *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 1440–1448, Santiago.

[15] REN Shaoqing, HE Kaiming, GIRSHICK Ross and SUN Jian (2015),” Faster R-CNN: Towards real-time object detection with region proposal networks”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.39 pp. 1137–1149.

[16] YASEEN Muhammad (2023), “What is YOLOv8: An in-depth exploration of the internal features of the next-generation object detector”, *National University of Computer and Emerging Sciences*, DOI: 10.48550/arXiv.2408.15857.

[17] YASEEN Muhammad (2023), “What is YOLOv9: An in-depth exploration of the internal features of the next-generation object detector”, *National University of Computer and Emerging Sciences*, DOI:10.48550/arXiv.2409.07813.

[18] WANG Ao, CHEN Hui, LIU Lihao, CHEN Kai, LIN Zijia, HAN Jungong and DING Guiguang (2024), “YOLOv10: Real-Time End-to-End Object Detection”, DOI: 10.48550/arXiv.2405.14458.

`

[19] ZZHANG Hong and ZHANG Sicong (2021), “A YOLOv5s-SE Model for Object Detection in X-ray Security Images”, *2021 International Conference on Control, Automation and Information Sciences (ICCAIS)*, pp. 626-631, Xi’an.

[20] KUMAR R. Senthil, A Syed Musthafa, ANBARASAN Balaji, KUMAR G. Singh and KUMAR Manikandaprabu P. (2022), “Recursive CNN Model to Detect Anomaly Detection in X-ray Security Image”, *2022 2nd International Conference on Innovative Practices in Technology and Management (ICIPTM)*, pp. 742-747, Gautam Buddha Nagar.

[21] GAUS Yona Falinie A., BHOWMIK Neelanjan and BRECKON Toby P (2019), “On the use of deep learning for the detection of firearms in X-ray baggage security imagery”, *IEEE International Symposium on Technologies for Homeland Security (HST)*, pp. 1-7, Woburn.

[22] AKCAY Samet and BRECKON Toby P. (2017), "An evaluation of region-based object detection strategies within X-ray baggage security imagery", *2017 IEEE International Conference on Image Processing (ICIP)*, pp. 1337-1341, Beijing.

[23] LIU Jinyi, LENG Xiaxu and LIU Ying (2019), “Deep convolutional neural network based object detector for X-ray baggage security imagery”, *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*, pp. 1757-1761, Portland.

[24] CHATURVEDI Kunal, BRAYTEE Ali, VISHWAKARMA Dinesh Kumar, SAQIB Muhammad, MERY Domingo and PRASAD Mukesh (2021), “Automated threat objects detection with synthetic data for real-time X-ray baggage inspection”, *2021 International Joint Conference on Neural Networks (IJCNN)*, pp. 1-8, Shenzhen.

[25] MITHAL Aditya, BASER Manit and DHIRAJ (2020), “Automatic threat detection in baggage security imagery using deep learning models”, *2020 IEEE 15th International Conference on Industrial and Information Systems (ICIIS)*, pp. 180-185, Rupnagar.

[26] JAIN Deepak Kumar and DHIRAJ (2020), “An evaluation of deep learning-based object detection strategies for threat object detection in baggage security imagery”, *Pattern Recognit Lett*, Vol 120., pp. 112-119.

`

[27] ZHOU Cheng, XU Hui, YI Bicai, YU Weichao and ZHAO Chenwei (2021), "X-ray security inspection image detection algorithm based on improved YOLOv4", *2021 IEEE 3rd Eurasia Conference on IOT, Communication and Engineering (ECICE)*, pp. 546-550, Yunlin.

[28] WANG Boying, ZHANG Libo, WEN Longyin, LIU Xianglong and WU Yanjun (2021), "Towards real-world prohibited item detection: A large-scale X-ray benchmark", *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5392-5401, Los Alamitos.

[29] KIM Hyo-Young, CHO Sung-Jin, BAEK Seung-Jin, JUNG Seung-Won, KO Sung-Jea (2021), "Learning-based image synthesis for hazardous object detection in X-ray security applications", *IEEE Access*, Vol. 9, pp. 135256-135265.

[30] TRIPATHY Saswati Soumya, BARIK Laxmipriya, SAHU Prajnya Paramita and NAIK Haraprasad (2021), "Automatic threat detection using deep neural networks", *2021 19th OITS International Conference on Information Technology (OCIT)*, pp. 66-71, Bhubaneswar.

[31] GÁLVEZ Alex, DADIOS Elmer P., VICERRA Rhandley D., BANDALA Argel A. and MANINGO Junrie (2019), "YOLO-based threat object detection in X-ray images", *2019 IEEE 11th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (HNICEM)*, pp.1-5, Laoag.

[32] AKÇAY Samet, KUNDEGORSKI M.E., DEVEREUX M. and BRECKON T.P. (2016), "Transfer learning using convolutional neural networks for object classification within X-ray baggage security imagery", *2016 IEEE International Conference on Image Processing (ICIP)*, pp. 1057-1061, Phoenix.

[33] BHOWMIK Neelanjan, GAUS Yona Falinie A. and BRECKON Toby P. (2021), "On the impact of using X-ray energy response imagery for object detection via convolutional neural networks", *2021 IEEE International Conference on Image Processing (ICIP)*, pp. 1224-1228, Anchorage.

[34] GOODMAN Eric, CHIEN Tiffany and MORRIS Trevor (2018), "Convolutional neural networks for automatic threat detection in security X-ray images", *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pp. 285-292, Orlando.

`

[35] BAŞTAN Muhammed, YOUSEFİ Mohammad Reza and BREUEL Thomas M. (2011), "Visual Words on Baggage X-ray Images" *In, Computer Analysis of Images and Patterns*, Eds. Pedro Real, Daniel Diaz-Pernil, Helena Molina-Abril, Ainhoa Berciano and Walter Kropatsch, pp.360-363, Springer Berlin, Heidelberg.

[36] BENEKYCIUK Emil, DENKOWSKI Marcin and DMITRUK Krzysztof (2021), "Material classification in X-ray images based on multi-scale CNN", *Signal, Image and Video Processing*, Vol. 15, No. 7, pp. 1285-1293.

[37] AKCAY Samet, KUNDEGORSKI M.E., WILLCOCKS C.G., BRECKON T.P. (2018). "On using deep convolutional neural network architectures for object classification and detection within X-ray baggage security imagery", *IEEE Transactions on Information Forensics and Security*, Vol. 13, No. 9, pp. 2203-2215.

[38] YUAN Jianping and GUO Chuangxin (2018), "A deep learning method for detection of dangerous equipment", *2018 Eighth International Conference on Information Science and Technology (ICIST)*, pp. 159- 164, Cordoba.

[39] MALARVIZHI Subramani, RAJADUARI K., CHOUDHURY Siddhartha Dhar and TOPKAR Anita. (2020), "Evaluating one stage detector architecture of convolutional neural network for threat object detection using X-ray baggage security imaging", *International Information and Engineering Technology Association*, Vol. 34, No. 4, pp. 495-500.

[40] CAIRONG Zhao, LIANG Zhu, SHUGUANG Dou, WEIHONG Deng and LIANG Wang (2022), "Detecting overlapped objects in X-ray security imagery by a label-aware mechanism", *IEEE Transactions on Information Forensics and Security*, Vol. 17, pp. 456-467.

[41] DOMINGO Mery, ERICK Svec, MARCO Arias, VLADIMIR Riffo, JOSE M. Saavedra and SANDIPAN Banerjee (2017), "Modern computer vision techniques for X-ray testing in baggage inspection", *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, Vol. 47, No. 4, pp. 678-689.

[42] ZHAO Cairong, ZHU Liang, DOU Shunguang, DENG Weihong and WANG Liang (2022), "Detecting overlapped objects in X-ray security imagery by a label-aware mechanism", *IEEE Transactions on Information Forensics and Security*, Vol. 17, pp. 1501-1513.

`

[43] MIAO Caijing, XIE Lingxi., WAN Fang, SU Chi, LIU Hongye, JIAO Jianbin and YE Qixiang. (2019), "SIXray: A large-scale security inspection X-ray benchmark for prohibited item discovery in overlapping images", *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3763-3772, Long Beach.

[44] WANG Boying, LIBO Zhang, LONGYIN Wen, XIANGLONG Liu, and YANJUN Wu. (2021), "Towards real-world prohibited item detection: A large-scale X-ray benchmark", *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 3170-3192, Montreal.

[45] WEI Yanlu., TAO Renshuai., WU Zhangjie., MA Yuqing., ZHANG Libo and LIU Xianglong. (2020), "Occluded prohibited items detection: An X-ray security inspection benchmark and de-occlusion attention module", *MM '20: Proceedings of the 28th ACM International Conference on Multimedia*, pp. 138-146, Seattle.

[46] MERY Daniel, VICTOR Riffo, UWE Zscherpel, GONZOLA Mondragón, IGNACIO Lillo, IVAN Zuccar, HORACIO Lobel, and MAURICIO Carrasco (2015), "GDXray: The database of X-ray images for nondestructive testing", *Journal of Nondestructive Evaluation*, Vol. 34, pp. 1-14.

[47] SOKOLOVA Marina and GUY Lapalme. (2009), "A systematic analysis of performance measures for classification tasks", *Information Processing & Management*, Vol. 45, pp. 427-437.

[48] EVERINGHAM Mark, LUC Van Gool, CHRISTOPER. Williams, WINN John, and ZIESSERMAN Andrew (2010), "The PASCAL Visual Object Classes (VOC) Challenge", *International Journal of Computer Vision*, Vol. 88, pp. 303-338.

[49] LIN Tsung-Yi, MAIRE Micheal, BELONGIE Serg and GIRSHICK Ross (2014). "Microsoft COCO: Common Objects in Context", *In, Computer Vision – ECCV 2014*, Eds. Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár and C. Lawrence Zitnick, pp.740-755, Springer, Zurich.

[50] GOODFELLOW Ian, BENGIO YOSHUA and COURVILLE Aaron (2016), *Deep Learning*, MIT Press, ABD.