Available online at www.ijournalse.org

**Emerging Science Journal**

(ISSN: 2610-9182)

Vol. 10, No. 1, February, 2026

# Metaheuristic Hyperparameter Optimization and Explainable Deep Learning for Baggage Threat Detection

Andino Maseleno [1], Miftachul Huda [2], Ahmad Fudholi [3], Chotirat Ann Ratanamahatana [1*]

[1] *Department of Computer Engineering, Faculty of Engineering, Chulalongkorn University, Thailand.*

[2] *Faculty of Human Sciences, Universiti Pendidikan Sultan Idris, Malaysia.*

[3] *Pusat Pengajian Citra Universiti, Universiti Kebangsaan Malaysia, Bangi, Selangor 43600, Malaysia.*

**Abstract**

The American Statistical Association reports that Bangkok, the capital and largest city of Thailand, holds the top spot as the most visited city worldwide in 2023. X-ray imaging for security screening plays a crucial role in upholding transportation security by detecting a diverse range of threats or prohibited items carried by passengers. This study introduces an advanced deep learning model leveraging YOLOv8, renowned for its enhanced efficiency in automating baggage detection processes. To enhance the model's hyperparameters and adjust them finely during the training process using the baggage dataset, the system utilized a metaheuristic optimization algorithm known as Evolutionary Genetic Algorithm, which is based on evolutionary principles. Incorporating explainable artificial intelligence techniques such as Local Interpretable Model-Agnostic Explanations (LIME) and Gradient-weighted Class Activation Mapping (Grad-CAM) allows for visual interpretation of predictions, aiding operators in utilizing the model effectively. We trained and tested the baggage dataset, which included 8,312 images and five classes: gun, knife, pliers, scissors, and wrench. The YOLOv8 model achieved the following metrics for the detection of prohibited objects in baggage inspection: an overall precision of 90.5%, recall of 83.3%, mAP50 of 91.3%, and mAP50-95 of 67%. The proposed method can fully automate the recognition of prohibited objects during baggage inspection. This approach is beneficial for designing an integrated, automatic, and non-destructive X-ray image-based classification system.

**Keywords:**
Deep Learning;
Baggage Detection;
Explainable Artificial Intelligence;
Hyperparameter Optimization.

**Article History:**

| | | | |
|---|---|---|---|
| **Received:** | 03 | September | 2025 |
| **Revised:** | 09 | December | 2025 |
| **Accepted:** | 21 | December | 2025 |
| **Published:** | 01 | February | 2026 |

## 1- Introduction

According to the American Statistical Association, the most visited city in the world for 2023 was Bangkok in Thailand [1]. With the global increase in passenger numbers, ensuring the security and safety of travelers has become a paramount concern. Among all the safety measures available at public places, X-ray screenings hold a top priority [2]. Detecting illicit items concealed within luggage presents a significant security challenge, as it poses a tangible threat to public safety [3]. On a daily basis, travelers in Thailand undergo thorough inspections for prohibited and dangerous items. Utilizing X-ray technology for security screenings facilitates the swift detection of metallic objects such as firearms [4]. At present, such applications are commonly utilized either to support baggage inspection personnel or to streamline the process through automation [5]. The scanning devices utilized in this context examine a range of items like packages, suitcases, parcels, and bags by projecting their images onto a screen linked to or built into the device. This projection relies on the X-ray radiation to be absorbed by the items inside. The resulting image employs pseudo-

* **CONTACT**: chotirat.r@chula.ac.th

**DOI:** http://dx.doi.org/10.28991/ESJ-2026-010-01-06

colors to represent density and atomic characteristics, thereby creating a visual depiction that is markedly distinct from natural optical imagery [6]. Given that the images are X-ray-generated, they encapsulate details regarding the faint shadowy patterns originating from the objects beneath the surface [7]. The task of scrutinizing these images for detecting prohibited items poses challenges stemming from various factors, including time constraints, fatigue resulting from demanding work schedules, the sheer volume of cluttered baggage, the intensity of aviation traffic, the labor-intensive nature of the task, or simply a shortage of personnel with the requisite expertise [8]. Operators are required to sustain an undivided focus on the screen, resisting distractions, a task of considerable complexity. Thus, ongoing research aims to devise methodologies for executing such demanding operations with heightened efficiency and diminished error margins. There have been considerable advancements in utilizing image processing technology for X-ray examinations in luggage screening inspections.

The YOLO (You Only Look Once) deep learning model stands as a cutting-edge, real-time object detection method [9]. This approach elegantly reframes object detection as a regression challenge, harnessing a unified deep neural network to seamlessly execute the entire detection process. In contrast, previous deep learning models such as Region-based Convolutional Neural Networks (RCNN) delineate the realization process into two discrete steps [10]. In the primary phase, myriad region proposals are generated, followed by a meticulous assessment of each proposal to ascertain its classification as either foreground or background. In the second stage, classification hinges on the characteristics defining the foreground. YOLO bypasses the necessity for iterative examination of an image (involving the extraction of numerous region proposals) to derive a classification outcome; instead, a singular pass suffices. It promptly produces both the coordinates and the probability distribution for each category in a unified operation.

The YOLO algorithms represent a suite of end-to-end deep learning models tailored for rapid and real-time object detection. Like YOLOv8 and its counterparts, these algorithms necessitate the calibration of model-specific hyperparameters prior to embarking on the training phase. Training the model with these parameters significantly enhances the network's behavior and performance [11]. Frequently, users rely on their experiential knowledge to initialize hyperparameters, yet even so, attaining the optimal outcome remains elusive. Within the realm of brute-force hyperparameter optimization, especially in complex, high-dimensional search spaces, reaching the pinnacle of hyperparameter configuration demands significant dedication and time investment. Indeed, due to such considerations, metaheuristic hyperparameter optimization becomes indispensable for minimizing computational costs and reducing user involvement [12]. Metaheuristic algorithms encompass a suite of techniques tailored to tackle intricate, expansive search spaces and non-convex optimization challenges, which include hyperparameter optimization. Here, we unveil the implementation of the genetic algorithm for optimizing hyperparameters within the YOLOv8-based algorithm.

One of the main obstacles in using deep learning-based solutions for image security is that these models are like black boxes. Operators often cannot fully understand why the machine makes a particular prediction. As a result, operators may harbor skepticism toward this emerging technology due to its opaque and inscrutable nature. A baggage detection system ought to exhibit transparency, comprehensibility, and explainability to earn confidence in operators. Ideally, it should elucidate the complete rationale behind its decisions to all stakeholders involved. This proposed initiative presents an elucidated deep learning-driven baggage detection system poised to discern luggage efficiently and instantly. To cultivate trust in the realm of baggage detection, we advocate the integration of sophisticated interpretable artificial intelligence techniques, notably LIME (Local Interpretable Model-Agnostic Explanations) [13] and Grad-CAM [14]. The structure of this paper is as follows: Section 2 offers a comprehensive overview of relevant literature. Methodological intricacies are delineated in Section 3, while Section 4 delves into the discourse surrounding outcomes gleaned from deep learning detection. Section 5 encapsulates concluding remarks and outlines avenues for future exploration. The favorable results of this investigation underscore the supremacy of the proposed approach over conventional methodologies.

## 2- Related Works

Research in the detection of threats based on objects in X-ray images used for luggage security has seen a steady rise, as illustrated in Table 1. Bhowmik et al. [15] attempted the detection of irregularities within X-ray security visuals. In Prior research, anomaly detection primarily focused on entire images or objects, overlooking potential anomalies hidden within specific image regions. Building on this insight, they proposed a novel approach, initially, grouping images to scrutinize anomalies across objects and their constituent parts. This involved employing segmentation techniques to identify anomalies via a secondary convolutional neural network, focusing on clustering at the object level. By leveraging a dataset of 14,964 X-ray images and employing knowledge transfer methods, they effectively identified anomalies like metal plates, screws, and knife blades concealed within electronic gadgets. Gaus et al. [16] delved into anomaly identification within X-ray luggage screening systems, a critical component of airport security. Their method relied on convolutional neural networks (CNNs) to autonomously identify anomalies in intricate X-ray security scans. They posited a two-step approach: initially categorizing liquid and electrical objects using advanced object detection framework such as region-based (R-CNN), mask-based (Mask R-CNN), and RetinaNet, and subsequently scrutinizing them for anomalies. This binary classification task distinguished between anomalies and non-anomalies, employing

standard CNN structures and fine-grained classification techniques. Their comprehensive experimentation, spanning object classification and anomaly detection, highlighted the complexity of the task while showcasing promising results.

Gaus et al. [17] explored security screening through X-ray imaging employing three deep network architectures. They assessed how well these models, optimized with Faster RCNN, Mask R-CNN, and RetinaNet frameworks, could generalize across different types of X-ray scanners. Given the scarcity of images depicting prohibited items, they investigated whether the networks could transfer their learning across scanners, treating it as an object detection problem. Their findings demonstrated improved generalization performance, particularly when dealing with specialized objects resembling explosives, thus highlighting the effectiveness of their approach. Liang et al. [18] investigated the automated identification and localization of dangerous substances to aid Transportation Security Management. Collaborating with the Transportation Security Administration (TSA), they created different deep convolutional architectures designed for detecting objects, trained on an extensive dataset covering various content and prohibited substances. These models were integrated into the Rapiscan 620DV scanner, resulting in a functional prototype capable of real-time operation. Their study emphasized the combined utilization of advanced neural network methods for feature extraction and subsequent object detection, utilizing techniques like SSD (Single Shot Detection), Faster-RCNN, Inception, ResNet, and InceptionResNet. Miao et al. [19] conducted investigation into detecting banned objects within X-ray imagery, aiming to advance research in this area. They compiled SIXray, a vast dataset comprising one million real-world instances, surpassing existing ones by a significant margin. This dataset reflects real-world scenarios, with about 1% of cases featuring forbidden items. To address the challenge of overlapping and redundant objects in X-ray imagery, they unveiled an algorithm to systematically refine mid-level features, effectively reducing redundant information. The iterative algorithm's sluggishness necessitates a single forward and feedback process, incorporating a cutoff mechanism. To rectify the significant data imbalance between positive and negative classes, a novel loss function was introduced. Evaluations across various popular network frameworks showcased impressive performance in both classification and localization tasks.

**Table 1. Related works**

| Publication Year | Author | Problem | Dataset | Method |
|---|---|---|---|---|
| 2019 | Bhowmik et al. [15] | Anomaly Detection | Manually Collected Dataset using 2D XRay Scanner | Using Mask-RCNN for region-of-interest segmentation, superpixel methods for sub-component analysis, and fine-grained CNN for classification |
| 2019 | Gauss et al. [16] | Object Segmentation | Manually Collected Dataset using single-view conventional Xray imagery | Utilizing Mask-RCNN for region-of-interest segmentation, and employing CNN classification for anomaly detection. |
| 2019 | Gauss et al. [17] | Object Detection | Durham Dataset Full Three-class (Dbf3), SIXray10, Durham Adversarial Dataset (DAD) | Investigating the adaptability across diverse X-ray scanners through the utilization of F-RCNN. |
| 2019 | Liang et al. [18] | Object Detection | Manually Collected Dataset | Among the array of detection algorithms explored, F-RCNN coupled with Inception ResNet v2 emerges as the top performer. |
| 2019 | Miao et al [19] | Object Classification | SIXray | Balancing classes hierarchically through refinement. |
| 2020 | Chouai et al. [20] | Object Segmentation | Hi-Tech Detection Systems Society (HDTS) private database | A convolutional neural network coupled with an adversarial autoencoder. |
| 2020 | Wei & Liu [21] | Object Detection | Grima X-ray Dataset (GDXray) | A transfer learning approach that integrates both classification and localization tasks within the SSD network architecture. |
| 2021 | Yao et al. [22] | Object Detection | Manually Collected Dataset | The network employs an N-type encoding structure to expand its receptive field. |
| 2021 | Dumagpi & Jeong [23] | Object Detection and Segmentation | SIXray | The Faster R-CNN model is utilized to pinpoint potential threat areas within the X-ray security images. |
| 2022 | Ma et al. [24] | Object Segmentation | PIXray | A Dense De-overlap Attention Snake (DDoAS) model |
| 2022 | Chang et al. [25] | Object Detection | SIXray | The proposed method for detecting prohibited objects in X-ray baggage images combines the backbone FPN (Feature Pyramid Networks), baseline Faster R-CNN, hard-negative-sample selection scheme, and physical size constraint. |
| 2023 | Fang et al. [26] | Object Detection | SIXray | A Few Shot SVM (FSVM) |
| 2023 | Wei et al. [27] | Object Detection | SIXray | Integrate the softer non-maximum suppression (Softer-NMS) algorithm with Mask R-CNN, resulting in the Softer-Mask R-CNN model. |
| 2024 | Andriyanov [28] | Object Classification | Ulyanovsk Civil Aviation Institute | Using ArcFace Loss Function and Softmax with temperature activation function |
| 2024 | Belal et al. [29] | Object Detection | SIXray, CLCXray, and COMPASS-XP | Balanced DINO framework that incorporates the Focal-Augmented Loss into a DINO-based Vision Transformer. |
| 2025 | Khan et al. [30] | Object Segmentation | SIXray, GDXray, and PIDray | SegFormer transformers |
| 2025 | Nasim et al. [31] | Object Segmentation | SIXRAY, GDXRAY, and PIDRAY | Incremental learning framework |

Chouai et al. [20] introduced a cutting-edge system for detecting objects in X-ray scans by combining a deep convolutional network coupled with an adversarial autoencoder. Their method segments images into overlapping areas, discerning between organic and inorganic substances and identifying overlaps of different object types. They departed from conventional methods by employing a distinct clustering technique that categorizes pixels into six classes, effectively addressing overlaps between organic and inorganic substances. This approach was then compared against the latest semantic segmentation systems in deep learning, demonstrating its superiority. Five distinct deep learning algorithms, namely Fully Convolutional Networks (FCN), SegNet, U-Net, DeconvNet, and RedNet, were compared in the study. Wei & Liu [21] employed SSD (Single Shot Detection) networks for both tasks involving classification and localization, leveraging multitask transfer learning approaches to address the challenge of gathering X-ray image datasets with hazardous objects. They took cues from DenseNet for their transfer learning method, integrating DenseBlock and Transition layers into their model to mimic DenseNet's structure, but confined these layers to the transfer layer section. Their approach involved leveraging weight data from another domain during the initial allocation, with only the transfer layer portion of the network being updated during training. In continuation, the study emphasizes relearning information from the model obtained in the originating domain by integrating an additional filter layer into the SSD network. Results from experiments indicate that this approach exhibits superior transfer learning capacity within SSD networks when contrasted with conventional fine-tuning techniques.

Yao et al. [22] approached the identification of perilous objects by framing it as a semantic segmentation assignment and introduced a versatile model for automated detection of prohibited objects. Acknowledging the inherent challenges of semantic segmentation, particularly in terms of misclassification and segmentation accuracy, the model that suggested incorporates two key enhancements. Firstly, it adopts an N-encoding architecture to widen the network's coverage area, focused on reducing mislabeling. This encoding strategy differs from conventional methods in that it includes two down sampling stages followed by one up sampling stage. Second, to tackle the challenge of limited surface texture in X-ray screening imagery, they repurposed superficial semantic information to enhance segmentation accuracy, taking cues from DenseNet's attribute reuse approach. Meanwhile, Dumagpi & Jeong [23] pioneered a pixel-level analysis method, departing from the conventional classification at the image level classification and detection used in X-ray screening imagery. This novel approach integrates separate modules for object detection and segmentation tasks. In the initial stage of the process, object detection was conducted using advanced deep learning techniques like Faster R-CNN, YOLOv3, SSD, and RetinaNet. Subsequently, in the following stage, the images were segmented into object and background components at the pixel level, utilizing methodologies, for instance DeepLabV3, PAN, MA-Net, PSPNet, U-Net, and FCN.

Ma et al. [24] explored convolutional neural network (CNN) segmentation techniques for X-ray screening imagery. They curated an image dataset comprising numerous overlapping cases for analysis. Subsequently, employing image dataset, they conducted segmentation using a DeepSnake-based approach. In contrast to DeepSnake, their approach implemented a vertex-based attention mechanism, weighing each vertex differently. Called DDoAS, their method comprises three modules: Dense De-overlap Module (DDoM), Attention Deforming Module (ADM), and One-to-One Fusion Module (O2OFM). Through DDoM, they employed weighted filtering to remove extraneous data originating from overlap and background. In the One-to-One Fusion Module, their aim was to retain edge data. ADM, on the other hand, involved acquiring weighted contour information using DeepSnake. They conducted a comparative analysis of this method against vanilla models to assess its effectiveness. Chang et al. [25] tackled the challenge of object detection in X-ray screening imagery using a dual-stage network. This approach aims to mitigate false alarms by taking into account the physical dimensions of restricted goods. Additionally, they employed hard negative sample selection to alleviate the effectiveness degradation resulting from the disparity in the distribution of positive and negative samples. This approach, they detect objects by segmenting the image containing negative samples, aiming to reduce the influence of irrelevant parts. They tested their approach leveraging the FPN and Faster R-CNN methods on the SIXray and OPIXray datasets, taking into account physical size and employing hard negative selection.

Fang et al. [26] developed a Few Shot SVM (FSVM) tailored for intricate distributed X-ray screening imagery containing prohibited objects. The FSVM is adept at detecting potential risks in X-ray luggage even with a limited number of samples. To enhance recognition accuracy, an end-to-end trainable SVM module is integrated within the object detection network. This enables the propagation of knowledge under supervision during back propagation, resulting in an improved embedding space for few-shot tasks. They evaluated four widely employed few-shot object detection (FSOD) methods to integrate few-shot learning techniques into X-ray threat detection. These methods include meta-learning, transfer learning, and pretrained transfer learning approaches. They presented results for both 10-shot and 30-shot few-shot samples using the dataset for X-ray luggage security SIXray. Wei et al., [27] introduced a manual segmentation technique to annotate the SIXRay dataset with pixel-level semantic information regarding hazardous objects. Additionally, they devised a technique for combining X-ray security inspection images to enhance positive samples effectively. This approach facilitates the quick generation of positive sample images by employing geometric transformation and utilizing HSV features extracted from X-ray images. Moreover, in a bid to enhance identification precision, particularly for hazardous objects positioned close together or overlapping, they introduced a hybrid approach by combining the algorithm designed for detecting targets, specifically Softer-NMS, with Mask R-CNN. This amalgamation, dubbed Softer-Mask R-CNN, outperforms the original model (Mask R-CNN) by 3.4% in accuracy (mAP) and by 6.2% when synthetic data is incorporated.

Andriyanov [28] used ArcFace Loss Function and Softmax with temperature activation function. Belal et al. [29] used balanced DINO framework that incorporates the Focal-Augmented Loss into a DINO-based Vision Transformer.

It focuses on fixing the common problem of class imbalance—where non-threat images overshadow rare threat cases—by boosting both algorithmic weighting and data augmentation strategies. Khan et al. [30] introduced SegFormer transformers for finding hidden threats in X-ray baggage images. Nasim et al. [31] proposed an incremental learning framework that enables the model to progressively learn new threat categories while retaining previously acquired knowledge.

Based on previous research data, it was discovered that there had been no prior studies on detecting baggage leveraging a deep learning model grounded on YOLOv8. Therefore, the study objectives are as follows:

- Utilize deep learning models like YOLOv8 to achieve precise baggage detection.
- Employ metaheuristic optimization for hyperparameter tuning process.
- Employ LIME and Grad-CAM techniques to improve model interpretability, facilitating a deeper understanding of the factors influencing model decisions and fostering confidence and trust in model usage.
- Discuss strategies for real-time deployment and accurate detection using the models.

## 3- Research Methodology

We utilize YOLOv8 object detection algorithms to identify threat objects. We focus on detecting five classes of harmful items—guns, knives, pliers, scissors, and wrenches—using the SIX-ray database. A key challenge we face with the dataset comprises the proportion of harmful to non-harmful objects, which is approximately 1:10 in the SIX-ray database. Furthermore, we conducted training, validation, and testing of all YOLO models using the same platform, specifically through Jupyter Notebook on Google Collaboratory (Google Colab). The machine learning model prototypes can be created from this platform by users, such as Graphics Processing Units (GPUs) and Tensor Processing Units (TPUs) [22]. All YOLO models underwent training and testing using Nvidia Tesla T4 GPUs with 16 GB of Graphics Double Data Rate (GDDR6) memory and Compute Unified Device Architecture (CUDA) graphic features. Figure 1 shows proposed methodology for baggage detection.

### *3-1- Datasets*

The SIXray dataset [19] is employed for identifying prohibited items in X-ray security images, sourced from various subway stations. This dataset was curated by the Pattern Recognition and Intelligent System Development Laboratory, located at the University of Chinese Academy of Sciences, conducted the research.

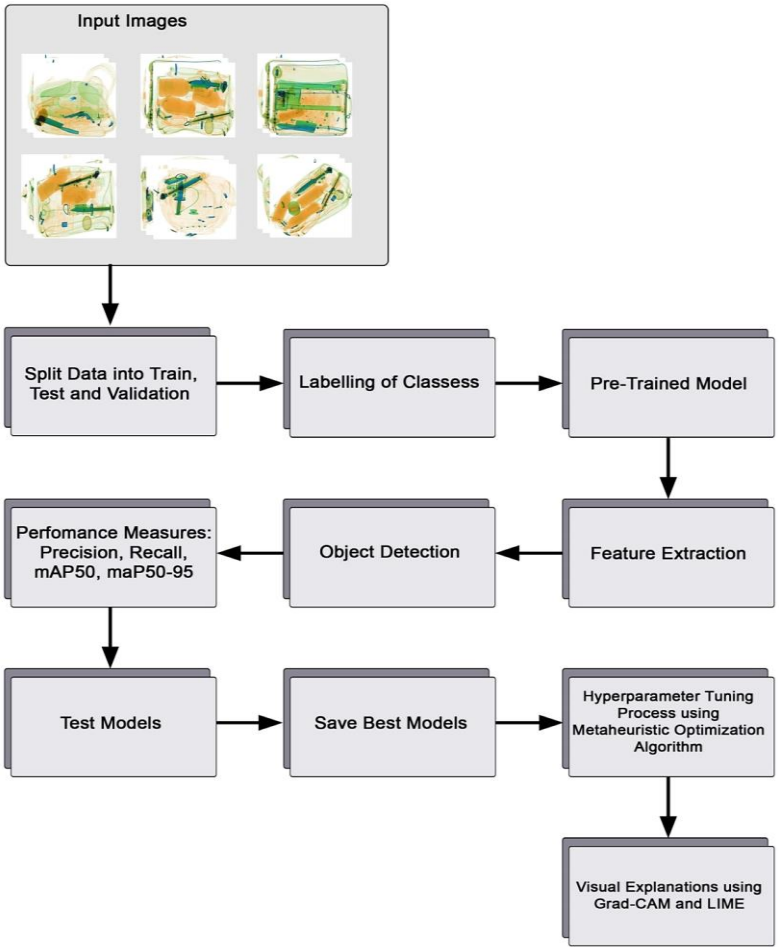


**Figure 1. Proposed methodology for baggage detection**

The images are obtained through a Nuctech dual-energy X-ray scanner, showcasing the common array of items found in everyday commercial baggage and parcels. The dataset encompasses objects with diverse scales, viewpoints, and often overlapping configurations, rendering it well-suited for applications requiring real-time classification, detection, and segmentation. Figure 2 shows examples of SIXray images. Figure 3 shows examples of SIXray negative images. In total, there are 8,823 images publicly accessible from SIXray. Of the 8,312 images in total, 70% (5,818 images) were utilized for training purposes, 20% (1,663 images) for validation, while the remainder 10% (831 images) for testing purpose the YOLOv8 network. The allocation of images among the datasets used for training, validation, and testing sets is shown in Figure 4. To make sure the dataset works well with the YOLOv8 model, it was converted into YOLO labeling format using Roboflow.

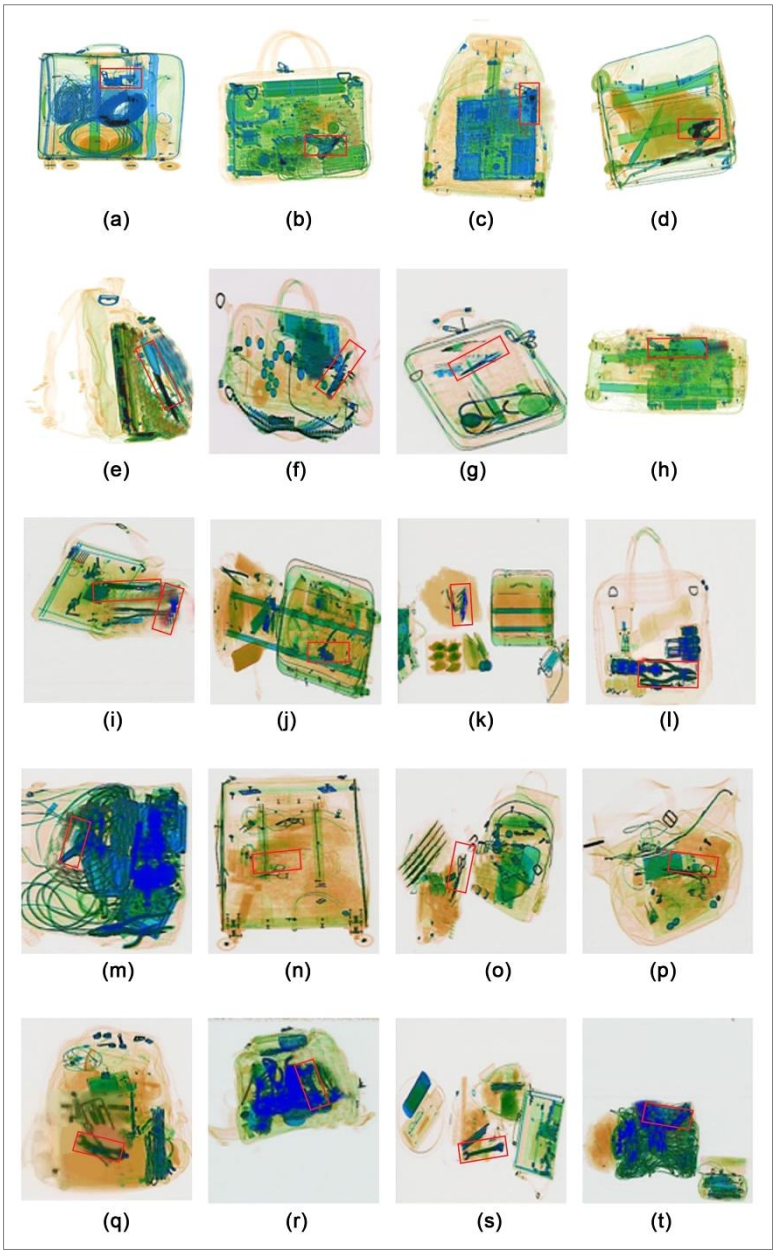


**Figure 2. Examples of SIXray images. Images (a) – (d): gun; images (e) – (h): knife; images (i) – (l): pliers; images (m) – (p): scissors; images (q) – (t): wrench**

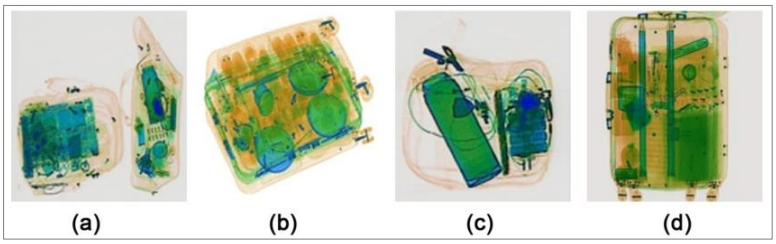


**Figure 3. Examples of SIXray negative images**

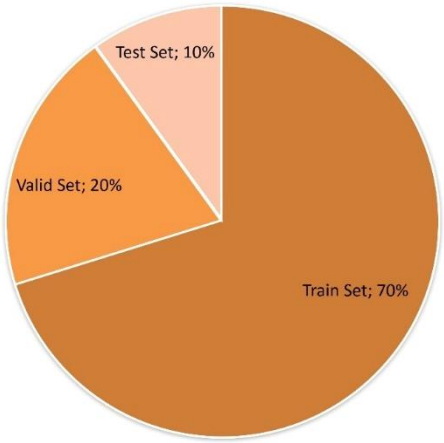


**Figure 4. Arrangement of dataset for training, validation, and testing purposes**

### *3-2- YOLOv8*

In computer vision, the suite of object detection algorithms known as YOLO has gained widespread popularity for its outstanding precision, compact size of the model, and ability to detect objects in real-time from images or videos. YOLO models are accessible to a broad range of developers as they can be trained using just one GPU. Additionally, machine learning specialists can utilize them on edge devices or in online storage at a reasonable cost. Nowadays, eight versions of the YOLO model have been developed, with each version introducing its own improvements and enhancements [32]. YOLOv8 stands out as the cutting-edge model in the YOLO series, achieving a fine equilibrium between precision in recognition and speed in detection. It represents a real-time and efficient single-stage object detection algorithm. YOLOv8 is an upgraded version of previous YOLO models, offering better performance and flexibility. It is designed to be fast and accurate. YOLOv8 uses a unique underlying architecture inspired by EfficientNet, a kind of network of interconnected neurons recognized for its capacity to attain high precision while using fewer parameters. This renders YOLOv8 demonstrates greater efficiency and faster than some other object detection models, all while upholding a commendable level of accuracy [33-35].

Published by Ultralytics in January 2023 [36], YOLOv8 is a product of the same company that brought us YOLOv5. Ultralytics specializes in developing and maintaining open-source software designed for computer vision activities. YOLOv8 offers five versions resized accordingly: nano, small, medium, large, and extra-large. It promotes various vision tasks including object detection, delineation, posture estimation, tracking, and categorization. Among them, YOLOv8x is the most accurate but least speedy, while YOLOv8 Nano is the fastest and most compact [37]. YOLOv8 distinguishes itself from YOLOv5 in several ways [38]: it uses the CSPLayer module, switches Conv in the backbone to a 3 × 3 Conv, eliminates Convs 10 and 14, modifies the initial 1 × 1 Conv in the bottleneck to a 3 × 3 Conv, and eliminates the presence of objects step by employing the decoupled head. YOLOv8 comprises three main components: the backbone, neck, and head. These components handle feature extraction, combining multiple features and generating prediction outputs, respectively. The YOLOv8 network design is illustrated in Figure 5.

In Figure 5, the numbers on the connecting line represent height × width × channel. CSPLayer_2Conv represent ‘CSPLayerWithTwoConv’ in MMYOLO repo. As the number of output channels varies in the last stage of different model sizes, r (ratio) is used in this figure for convenience, ‘last_stage_out_channels’ is used to control the number.

The backbone network composed of the layers involving convolutions is responsible for extracting features from the input image, enabling further processing and analysis. The CSPDarknet53 backbone network is adopted to YOLOv8, the upgraded version of Darknet53 that was used previously. While similar to YOLOv5's backbone, YOLOv8 introduces modifications to the CSPLayer module [39]. This module, known as the cross-stage partial bottleneck, comprises two convolutional layers, bolsters the precision of detection by amalgamating contextual information to high-level features. The feature extraction network focuses on capturing features at different scales from images to provide a comprehensive representation of the input data that is generated by the CSPLayer and SPPF modules. Additionally, the C2f module streamlines the system by reducing a singular convolutional layer derived from the original C3 module, resulting in a lighter model. Furthermore, this integrates YOLOv7 to the advantages of the ELAN architecture, effectively extending the gradient pathway through bottleneck components to acquire more comprehensive gradient flow data [40]. SPPF optimizes the network's layers by leveraging spatial pyramid pooling or SPP principles [41], reducing redundant operations and accelerating feature integration. The SPPF module processes output feature maps by pooling with various kernel sizes to blend the feature maps, before transmitting the results to the neck layer [42].

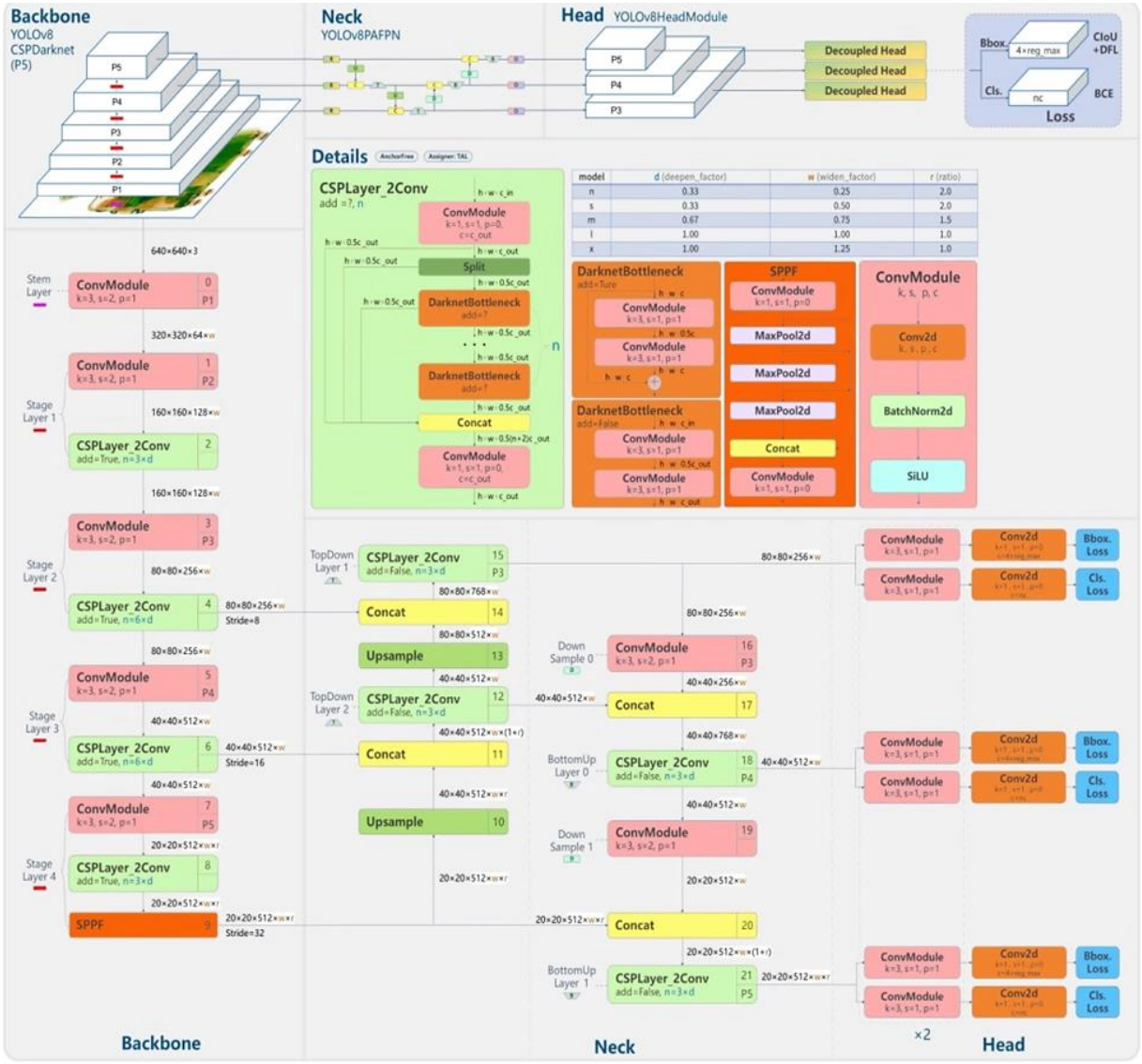


**Figure 5. YOLOv8 Architecture [35]**

The network's neck incorporates extra convolutional layers to enhance the feature representations yielded by the primary network architecture. YOLOv8 integrates a Spatial Attention Module (SAM) to enhance feature representation. Moreover, it adopts feature pyramid networks and neck path aggregation network structures, enhancing the feature of the network fusion capability by integrating features of varying levels. The multiscale fusion module strategically merges features from both the feature pyramid network [43] and path aggregation network [44]. By effectively integrating low-level features with reduced receptive areas and high-level features, it optimizes feature representation and enhances the detection capability of targets across a spectrum of scales.

The main network, comprised of some convolutional layers, produces predictions of bounding box positions and class likelihoods for every grid cell within the resulting feature component. In YOLOv8, the head adopts an anchor-free approach, departing from the anchor-based method. Instead of IoU matching or single-side scale assignment, it employs a task-specific assigner for matching positive and negative samples. Ultimately, it conducts various level predictions utilizing 8x, 16x, and 32x down-sampled features, ensuring accurate predictions for these three-target levels. YOLOv8 incorporates a SPP (Spatial Pyramid Pooling) module to adeptly capture object characteristics across a range of scales. The final predictions are generated by the output layer of YOLOv8. Employing a decoupled head in an anchor-free model, YOLOv8 autonomously manages objectivity, classification, and regression tasks. This design strategy enables each branch to concentrate solely on its designated task, thus enhancing the total precision of the model. The activation function for the objectivity score is the sigmoid function in YOLOv8 at the output layer, indicating the object likelihood being present within the bounding box prediction. Meanwhile, the likelihood class is determined using the function of softmax, indicating the probability of the belonging objects to each potential class. YOLOv8 implements advanced loss functions such as CIoU [45] and DFL [46] for bounding-box loss, while using binary cross-entropy to calculate the loss classification. The intricate loss functions have proven instrumental in improving the ability to detect objects, particularly in scenarios involving smaller objects.

### *3-3- Evaluation Metrics*

The efficacy of the YOLO algorithm in object detection is contingent upon numerous variables, including precision (P), recall (R), F1 score, average precision (AP), intersection over union (IoU), and the non-maximum suppression (NMS) threshold [35]. For the experiment, three pivotal performance measures for object detection duties were carefully selected: Precision, Recall, and mAP (mean Average Precision).

**1)** ***Precision***

Precision is the ratio of correctly identified objects to the total number of objects detected by the algorithm. It is computed using the following formula.

*Precision = True Positives / (True Positives + False Positives)*

A high precision value indicates how accurately the algorithm identifies objects to provide accurate outcomes and effectively discern negative samples. Precision, as a metric, places emphasis on both false positive and true positive outcomes. High thoroughness is achieved when the occurrence of instances of false positives is minimal, highlighting the algorithm's capability to precisely identify relevant objects while minimizing erroneous detections.

**2)** ***Recall***

Recall, a crucial metric, represents the proportion of objects identified by the algorithm compared to the actual number of objects present. It delineates the number of accurately predicted values from the total records for each class, thereby elucidating the model's proficiency in identifying positive samples. The calculation of recall is based on the following equation.

*Recall = True Positives / (True Positives + False Negatives)*

Recall, often referred to as the True Positive Rate, evaluates the algorithm's performance to capture all actual objects. A high recall value means the algorithm can effectively spot positive samples without missing genuine objects, thereby enhancing the accuracy of positive sample identification.

**3)** ***Average Precision***

Average Precision (AP) serves as a metric to assess the algorithm's object detection accuracy across various classes. It is computed by iteratively calculating precision and recall values for each class. Mean Average Precision (mAP) represents the average precision value across all detection categories and is a key indicator of the model's overall precision. The computation for mAP involves summing up precision values for all categories and dividing by the total number of categories.

### *3-4- Metaheuristic Hyperparameter Optimization*

Hyperparameters serve as parameters used to configure a neural network or to specify the algorithm used to minimize the loss function [47]. Crafting an optimal deep learning model entails exploring a multitude of possibilities. This endeavor, termed hyperparameter tuning, involves sculpting the ideal model architecture with a refined hyperparameter configuration. The meticulous adjustment of hyperparameters is acknowledged as a pivotal aspect in the creation of a potent deep learning model. Hyperparameters play a pivotal role in regulating the performance of applied techniques [48]. This is essential for refining the training of deep learning algorithms. The selection of hyperparameters is a defining feature of deep learning models, as it elucidates specific parameters responsible for various aspects such as memory management and computational complexity. Careful selection of hyperparameters can substantially bolster the efficacy of the model, following the selection of an appropriate dataset and model. Through meticulous adjustment of hyperparameters, the model's performance can be notably enhanced.

The hyperparameter tuning process varies across different deep learning algorithms owing to their diverse sets of hyperparameters [49]. Parameters play a pivotal role in convolution networks, governing the entire training process and exerting the type of loss function chosen can greatly affect how well the final detection model performs. However, for models utilizing a novel framework, fine-tuning parameters individually can prove time-consuming and ineffective. Additionally, improperly adjusted parameters may not accurately showcase the model's performance. Fine-tuning parameters can improve both recall and precision by setting an appropriate threshold, which is crucial for achieving an effective model [50]. Manual testing, a traditional approach for hyperparameter tuning, persists in the research conducted by graduate students. However, it necessitates a deep understanding of the utilized deep learning algorithms and their respective hyperparameter configurations [51]. Nevertheless, manual tuning proves ineffective for many problems due to various factors, such as the hyperparameters multitude, sophisticated models, lengthy evaluations, and intricate connections between hyperparameters. Those components have spurred increased research into techniques for automatically optimizing hyperparameters, often referred to as hyperparameter optimization [52]. The primary goal of

hyperparameters is to mechanize the process of tuning, streamlining the application of machine learning models to real-world issues for users [53]. This optimization process is essential for improving model performance and adaptability. Figure 6 shows the flowchart outlines the process of utilizing a genetic algorithm for hyperparameter optimization.

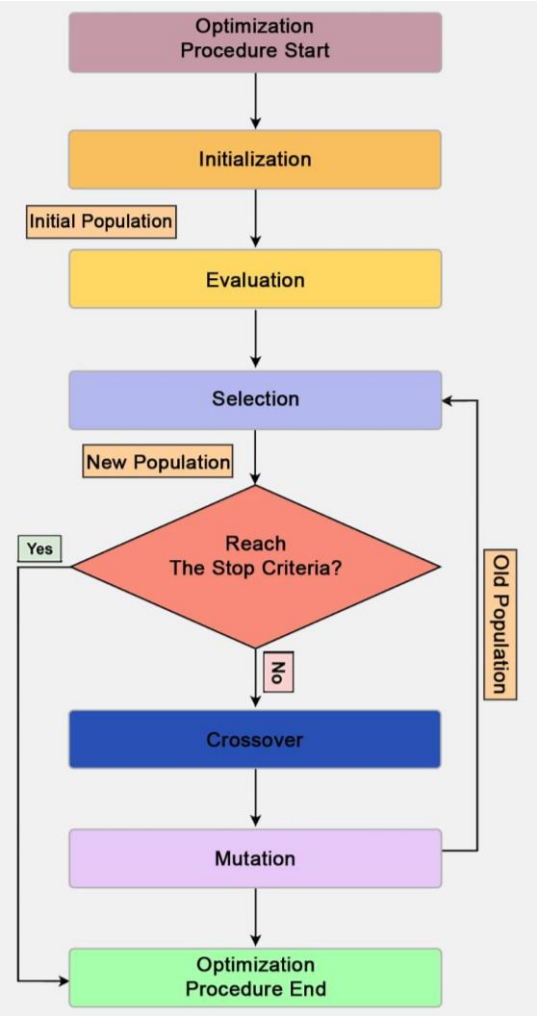


**Figure 6. The flowchart outlines the process of utilizing a genetic algorithm for hyperparameter optimization**

A genetic algorithm is a type of metaheuristic inspired by the principles of natural selection, falling under the broader category of evolutionary algorithms. It mimics the process of natural selection observed in biological evolution. The algorithm draws its inspiration from Darwin's theory of evolution, which suggests that species evolve over time due to competition among different variations within the species. Individuals with traits that enhance survival and reproduction are more likely to persist within the species, while those with detrimental traits are likely to diminish. This process, termed natural selection by Darwin, gradually alters the characteristics of the species, leading to its evolution over time [54]. Natural selection occurs when advantageous traits, encoded in the genetic makeup of individuals, are passed down to offspring and become more prevalent in subsequent generations. Traits conferring reproductive benefits gradually become more common over time, leading populations to better fit their environments and become more adapted. Unlike other evolutionary mechanisms, natural selection specifically drives adaptation [55].

Genetic algorithms excel as global optimization techniques capable of discovering near-optimal solutions throughout the entire search space. This stands in contrast to local optimization methods like grid search and random search, which have more limited scopes. Genetic algorithms outperform grid search and random search due to several key factors [56]. They are adept at handling large search spaces, where grid search becomes unwieldy and random search may not be the most effective due to its randomness. Their iterative refinement process strikes a balance between exploration and exploitation, gradually enhancing solutions over generations. By framing the task of selecting optimal hyperparameter values as an optimization problem and employing the genetic algorithm, we aim to find the best-performing solution.

The genetic algorithm provides a method for addressing optimization challenges, whether constrained or unconstrained. It iteratively enhances a population of potential solutions, seeking the most optimal solution within a vast search space. This algorithm follows several stages, including the creation of an initial population, evaluation of fitness, selection, crossover, and mutation. Initially, the genetic algorithm generates a population comprising randomly selected

solutions. The initial population, consisting of randomly generated individuals or genes (such as learning rates), have a vital role in determining the quality of solutions. This population typically comprises learning rates within a predefined range, often based on previous research findings. Fitness evaluation, which involves assessing the performance of all genes, is a critical step in the process. Here, the fitness of each gene is measured using metrics such as mAP of the model. Selection then identifies the top-performing genes based on their fitness scores, ensuring that only the best candidates progress to the next stage. Crossover, the subsequent step, involves combining the traits of the selected genes to create offspring with potentially superior characteristics. During this process, a chromosome is formed by merging the traits of two highly fit candidates. Mutation, akin to biological mutation, introduces alterations to chromosome values, thereby increasing diversity within the population. This phase aims to enhance the exploration of potential solutions. The decision to perform crossover and mutation is governed by predefined probabilities.

The genetic algorithm continues to iterate through these steps, training and evaluating the model, until reaching the upper limit of generations or iterations. The selection process decides which genes will undergo crossover or mutation. Genes are assigned a random float value between 0 and 1. If this value is below a set probability threshold, the gene is selected for crossover or mutation. In mutation, the gene's hyperparameter, like the learning rate, gets replaced with a new random value. Simple Arithmetic Crossover blends the attributes of two parent individuals based on a mixing ratio called alpha. This ratio, ranging from 0.00 to 1.00, determines the influence of each parent on the offspring. When a gene is chosen for crossover, it is split into two parts—head and tail—using a randomly generated value between 0 and the gene's length as the dividing point. The crossover operation involves using the tail side of each parent to generate a new offspring or genes. Optimal hyperparameters play a pivotal role in assessing the model's performance as they govern training the network across different stages. Using the best values for these variables enhances various aspects of model performance, including metrics like mAP, Precision, and Recall rate. The genetic algorithm outperforms conventional methods such as Random Search and Grid Search, particularly when confronted with many dimensions and unidentified correlations between dimensions [57].

## 4- Results and Discussion

In our process, all of YOLO models underwent training, validation, and testing by employing a uniform platform—Jupyter notebook in Google Collaboratory. These platforms facilitate the prototyping of deploying machine learning models on instruments such as Graphical Processing Units (GPUs) and Tensor Processing Units (TPUs).

The confusion matrix, as shown in Figure 7, shows clearly that our proposed system based on YOLOv8 achieves the best results.

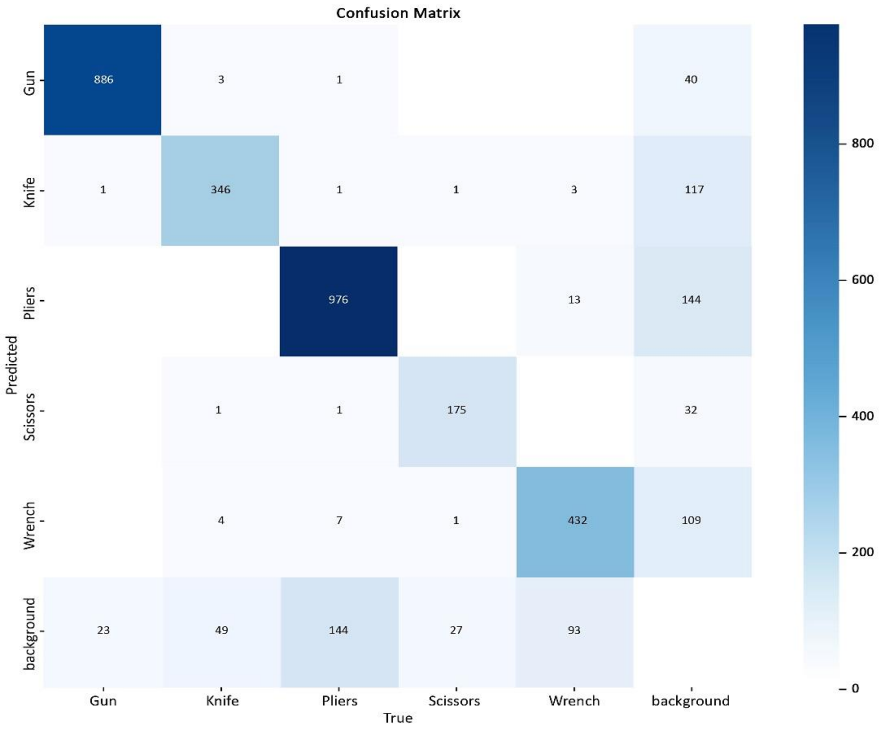


**Figure 7. Confusion matrix**

Figure 8 shows a graph obtained from training the YOLOv8 model. Throughout each epoch, the model of YOLO travels across every confining box in the dataset, fine-tuning its model variables derived from the designated loss

function and optimization method. Adding more epochs means the model of YOLO goes through the data source more times. But picking the right number of epochs is important. Too few might miss important features (underfitting), while too many might make the model too specific to the training data (overfitting). So, finding the right balance often takes some trial and error. In the training phase, the optimal epoch values fell between 16 and 22. Consequently, we set the epoch value to 25, aiming to maximize results within the constraints of limited training. Training 8,312 images over a span of 25 epochs required a specific time investment with anticipating the satisfactory outcomes. To ensure an equitable comparison among models, the batch size was kept consistent for all experiments.

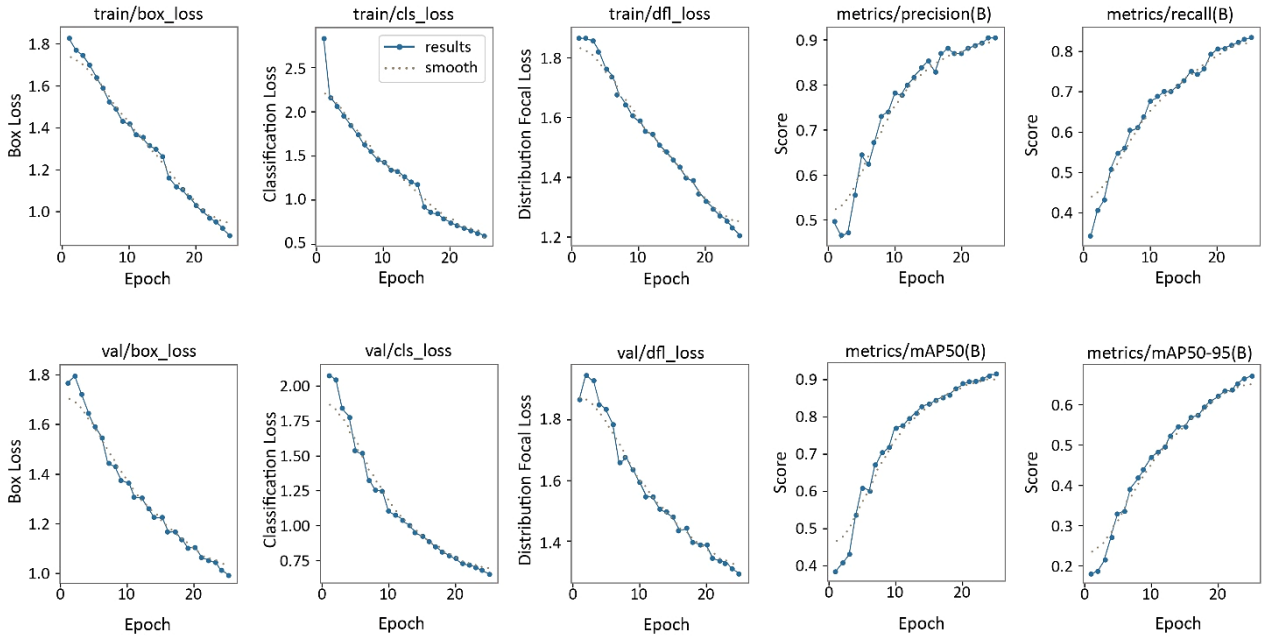


**Figure 8. Training graph of YOLOv8 model**

Table 2 shows precision, recall and mAP for all classes. Precision measures how many positive detections were correctly identified out of all the predicted positive detections. It indicates how well the algorithm avoids false positives. Recall computes the proportion of correctly identified positive detections out of all actual positive instances in the ground truth. It evaluates the capability of the algorithm to minimize false negatives. mAP is a commonly used metric in tasks involving the detection of objects. It calculates the average precision across different object categories and IoU thresholds, providing a holistic assessment of the algorithm's object detection precision. The precision scores for gun, knife, pliers, scissors, and wrench stand at 97.4%, 83.8%, 92%, 91.5%, and 87.8%, respectively. Overall, these numbers indicate a solid precision performance, averaging 90.5% across all five classes. When it comes to recall, the rates for gun, knife, pliers, scissors, and wrench are 95.8%, 80.4%, 81.9%, 84.3%, and 74.2%, respectively. This yields an overall recall of 83.3% for the entire dataset. In terms of mAP50, gun achieves an impressive 99%, followed by knife with 88.5%, pliers with 91.8%, scissors with 91.4%, and wrench with 86.1%. The overall mAP50 for all classes is a commendable 91.3%. Furthermore, considering mAP50-95, gun maintains a high 78.8%, while knife, pliers, scissors, and wrench achieve 61.5%, 67.4%, 66%, and 61.5%, successively, culminating in an overall mAP50-95 of 67% across all classes. Figure 9 presents graphs of the detection results.

**Table 2. Detection results**

| Class | Precision | Recall | mAP50 | mAP50-95 |
|---|---|---|---|---|
| All | 0.905 | 0.833 | 0.913 | 0.67 |
| Gun | 0.974 | 0.958 | 0.99 | 0.788 |
| Knife | 0.838 | 0.804 | 0.885 | 0.615 |
| Pliers | 0.92 | 0.819 | 0.918 | 0.674 |
| Scissors | 0.915 | 0.843 | 0.914 | 0.66 |
| Wrench | 0.878 | 0.742 | 0.861 | 0.615 |

Figure 9 presents the performance of an object detection model evaluated on different object classes using four key metrics: Precision, Recall, mAP50, and mAP50-95. Each class, including the overall performance labeled as "All," is represented with a group of four bars, making it easy to compare the metrics side by side. The vertical axis indicates the score values ranging from 0 to 1, where higher values represent better performance.

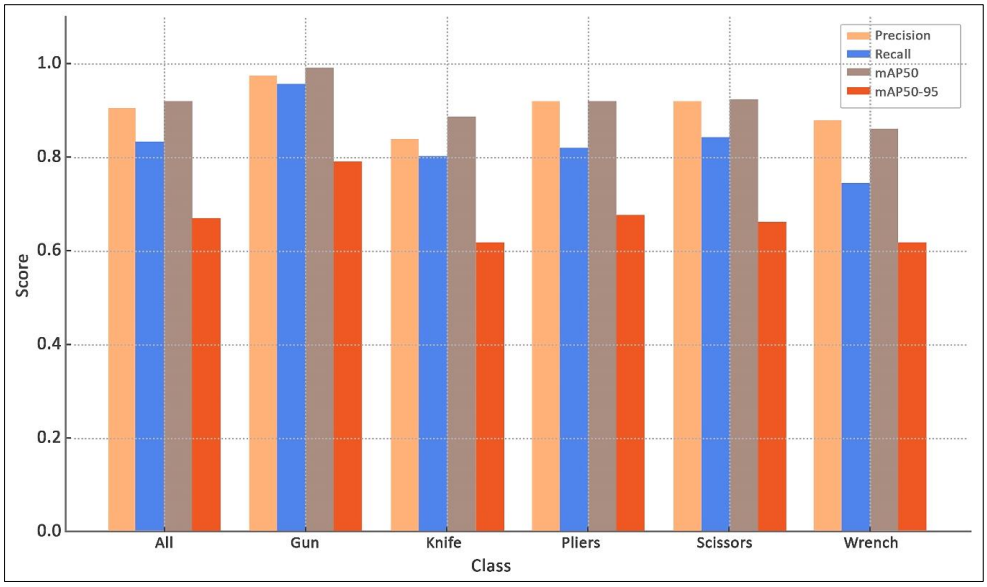


**Figure 9. Object detection evaluation metrics**

From the results, it is evident that the model performs exceptionally well on the Gun class, achieving very high Precision (0.97) and Recall (0.96). This means that almost all detected guns are correct and that most actual guns are successfully detected. The Gun class also records the highest mAP50 (0.99) and the strongest performance under the stricter evaluation of mAP50-95 (0.79). In contrast, the classes Knife and Wrench show weaker results, particularly in Recall and mAP50-95 (both around 0.615), indicating that the model struggles more with accurately detecting and localizing these objects. The Pliers and Scissors classes demonstrate relatively strong performance, with Precision and mAP50 values above 0.91, but their Recall and mAP50-95 values show moderate decreases. This suggests that while the model is confident when it detects these objects, it sometimes misses actual instances or struggles with precise localization at higher thresholds.

The aggregated "All" category shows that the model achieves good performance with Precision (0.91), Recall (0.83), and mAP50 (0.91). However, the stricter mAP50-95 metric is significantly lower (0.67), highlighting that while the model is effective at detection, its bounding box localization could be improved for tighter accuracy. In summary, the model is most reliable for detecting Guns, performs moderately for Pliers and Scissors, and requires improvement for Knife and Wrench. Figure 10 shows the output detection with YOLOv8.

Figure 10 presents the output of an object detection task using the YOLOv8 model on X-ray images of luggage. Each panel in the grid corresponds to a different luggage scan, where the model has identified and localized objects of interest by drawing bounding boxes around them. The detections focus on potentially dangerous or prohibited items such as knives, pliers, and wrenches, which are often subject to security screening in airport or checkpoint scenarios.

The bounding boxes are color-coded to indicate the type of object detected: knives are shown in pink, pliers in orange, and wrenches in yellow. Alongside each bounding box, the model provides a confidence score between 0 and 1, which reflects the probability that the identified object truly belongs to the predicted class. For example, a label reading "Pliers 0.8" indicates that the model is 80% confident that the detected object is a pair of pliers. These confidence levels vary across detections, with higher values showing strong certainty and lower ones suggesting ambiguity due to factors such as object overlap, occlusion, or similarity in shape to other items. The figure demonstrates YOLOv8's ability to handle cluttered environments, as luggage often contains numerous overlapping objects that make detection more difficult. Some images show multiple objects correctly detected within the same bag, such as pliers and wrenches together, highlighting the model's robustness in multi-object scenarios. However, variations in detection confidence also reveal the challenges of distinguishing items under complex conditions, such as when metallic parts overlap or when objects are partially obscured.

Figure 10 illustrates the practical application of YOLOv8 for automated threat detection in security screening. It highlights both the strengths of the model, including its high accuracy and speed in identifying multiple items, as well as its limitations, where certain detections may have lower confidence or potentially miss hidden items. This type of system could play an important role in assisting human operators by reducing workload and improving detection

efficiency in real-world safety and security environments. Figure 11 shows results of hyperparameter optimization of YOLOv8 with 100 epoch.

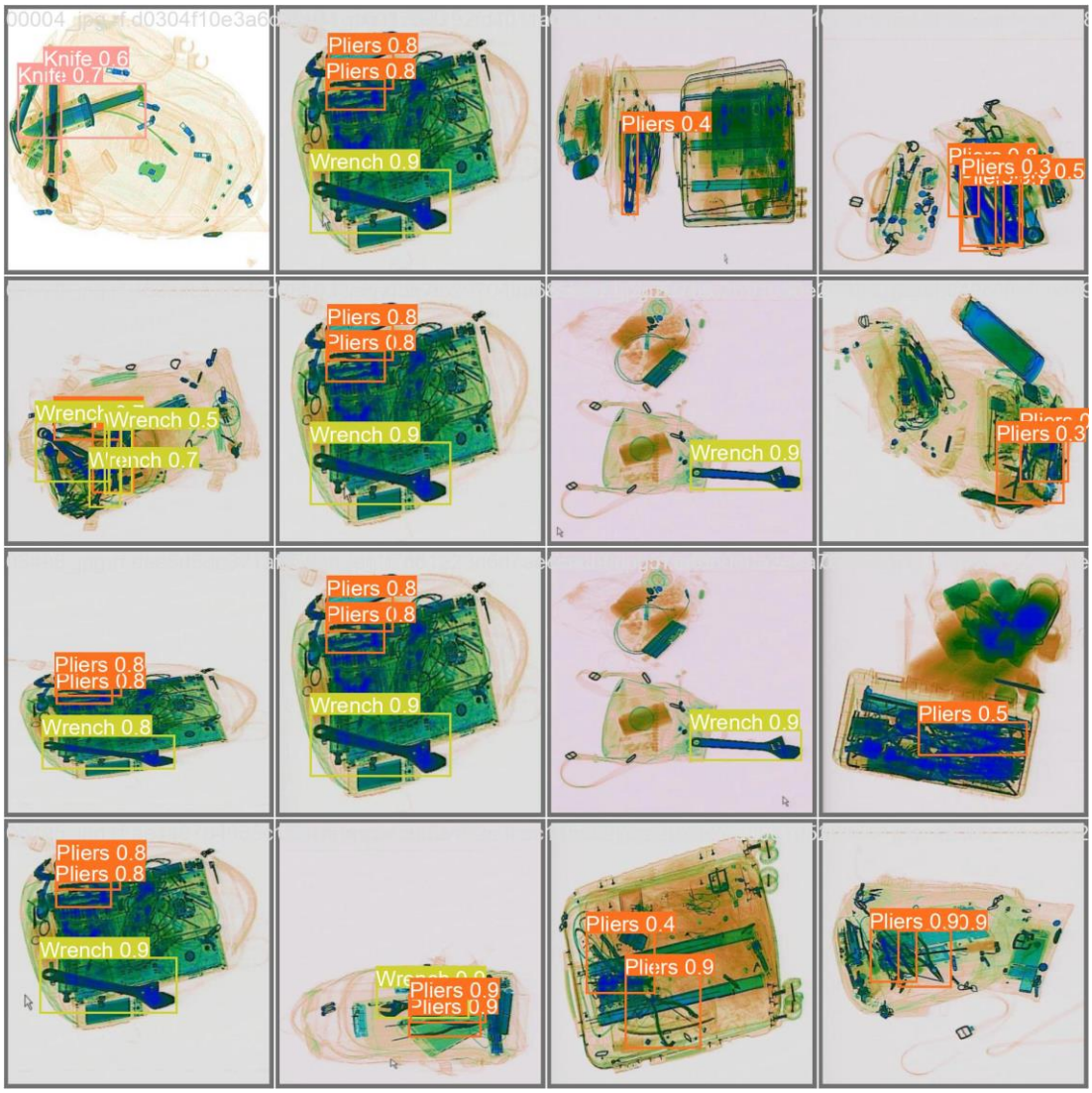


**Figure 10. The output of detection with YOLOv8**

From the Figure 11, we can see that a variety of training-related hyperparameters were optimized:

- Learning rate parameters (*lr0*, *lrf*), momentum, weight decay, and warmup settings control the optimization dynamics.
- Data augmentation hyperparameters (*hsv_h*, *hsv_s*, *hsv_v*, *degrees*, *translate*, *scale*, *shear*, *perspective*, *flipud*, *fliplr*, *mosaic*, *mixup*, *copy_paste*) influence how the training images are transformed to improve model generalization.
- Loss and label-related parameters (*box*, *cls*, *dfl*) balance contributions of box regression, classification, and distribution focal loss.

Some hyperparameters remained fixed at zero (e.g., *shear*, *perspective*, *flipud*, *mixup*, *copy_paste*), indicating that these transformations were not found beneficial under this optimization. Others, like *mosaic = 1*, *fliplr = 0.5*, *scale = 0.5*, *and hsv adjustments*, were actively used, suggesting that they contribute positively to improving generalization in this dataset. Similarly, learning rate (*lr0 = 0.01*) and momentum (*0.937*) align with typical YOLO training defaults but were validated as effective through this tuning. The figure demonstrates that hyperparameter optimization successfully identified suitable configurations for both training and data augmentation strategies in YOLOv8. This process enhanced the balance between precision, recall, and generalization, which is crucial for baggage threat detection where both accuracy and robustness to variations in input data are essential.

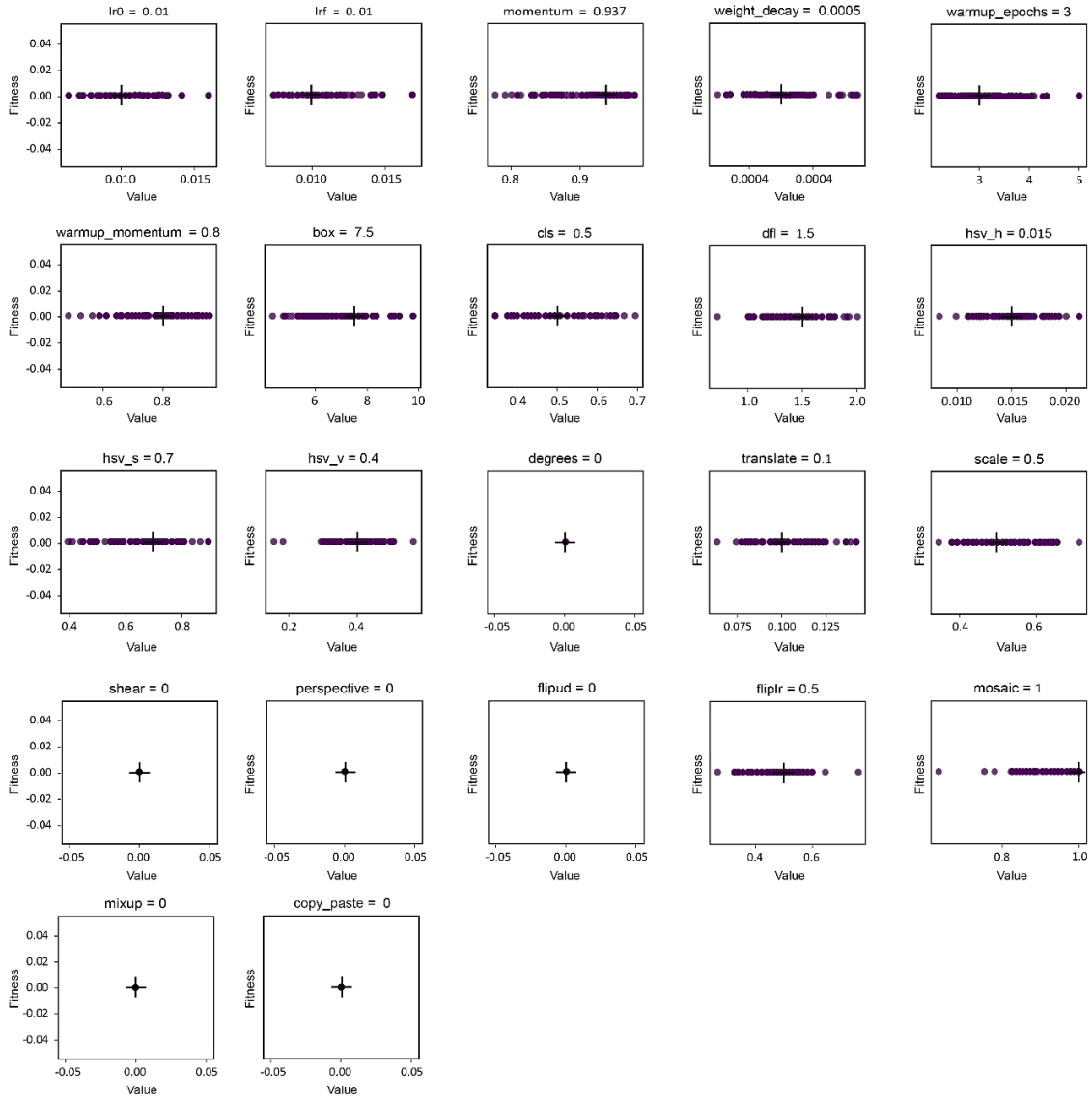


**Figure 11. Results of hyperparameter optimization of YOLOv8 with 100 epoch**

## Explainable Deep Learning

Given the black box approach of deep learning classifiers, employing visualization methods becomes crucial for grasping the features acquired by the convolutional neural network during training. By leveraging these insights, researchers can delve into additional tasks like refining optimization strategies and exploring alternative models. This approach aids in addressing issues like erroneous feature acquisition and overfitting, thereby enhancing the overall effectiveness and the model interpretability. We employed Gradient-weighted Class Activation Mapping (Grad-CAM) to generate visual explanations for the decisions made by CNN-based models, thereby improving transparency and interpretability. Grad-CAM leverages gradients from images as they propagate through the end of convolutional layer, creating a localization map that underscores the pivotal regions in the picture for predicting the concept [14]. By leveraging gradient information from the last convolutional layer, Grad-CAM assigns importance values to individual neurons, shedding light on their contribution to specific decisions of interest. Figure 12 shows Grad-CAM explanations.

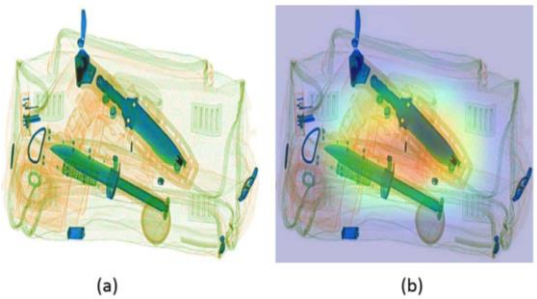


**Figure 12. Grad-CAM Explanations**

Figure 12 demonstrates the application of Grad-CAM to explain model predictions in the context of X-ray baggage screening. Grad-CAM is a visualization technique that highlights the regions of an image that contribute most to a deep learning model's decision, thereby making the model's reasoning process more interpretable. The figure consists of two subfigures: (a) the original X-ray scan, and (b) the Grad-CAM explanation overlay.

In subfigure (a), the raw X-ray image of a bag is shown. As is typical with security scanner imagery, the contents are color-coded based on material density, with metallic objects such as knives appearing in darker colors. This image serves as the input to the deep learning model, which is tasked with detecting prohibited items hidden inside luggage. In subfigure (b), the Grad-CAM heatmap is superimposed on the original X-ray scan. The heatmap uses a color spectrum (ranging from blue to red) to indicate the importance of different regions in the model's decision-making process. Warmer colors such as yellow and red highlight the areas that the model considered most relevant when predicting the presence of a knife. In this case, the highlighted regions are centered on the knives inside the bag, confirming that the model is focusing on the correct objects of interest rather than being distracted by irrelevant background items.

Figure 12 illustrates how Grad-CAM improves the transparency of deep learning models in baggage screening. By providing visual evidence of where the model directs its attention, Grad-CAM helps build trust in automated threat detection systems. Security operators can verify that the AI model bases its predictions on meaningful features, such as the actual shape and position of knives, rather than unrelated parts of the image. This combination of raw input and Grad-CAM visualization strengthens confidence in deploying AI for high-stakes security applications.

LIME technique, introduced recently, aims to approximate a locally linear model from a complex one [58]. This allows interpreting the weights of linear combinations as the feature importance score. Additionally, a classical technique called influence functions has been efficiently employed to identify the training data instances primarily accountable for a specific prediction output. The integration of these advanced computational methods with interactive visualization techniques holds significant potential for explainable deep learning. Yet, it remains a significant challenge in practical scenarios [59]. The visualization technique using LIME is widely employed to visualize the predictions made by the convolutional neural network. LIME operates by taking the input image data and introducing random perturbations to elucidate the mechanisms behind the predictions. The XAI visualization is based on the best-performing model. Within the field of machine learning, interpreting a deep learning model typically involves determining the scores that indicate feature importance. This entails recognizing the specific section of the input feature of a given data item influences the forecasting at the outcome of output layer and/or in triggers high activation of an internal layer or node. Researchers in the fields of machine learning and artificial intelligence have been working on innovative solutions to tackle this issue. Perturbation experiments [60] and methods based on saliency maps [13] have proven effective in proofing which parts of an input image are most influential in the model's final prediction. Figure 13 shows LIME explanations.

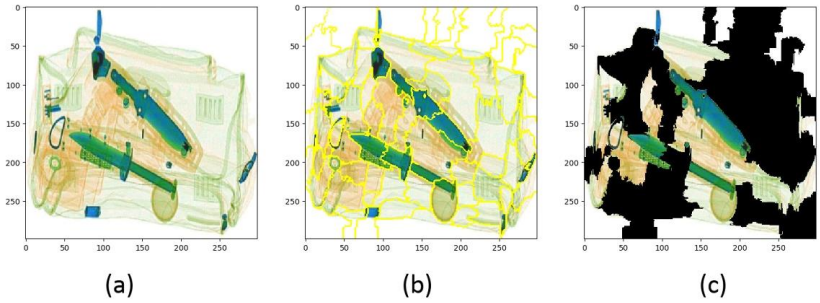


**Figure 13. Training explanations for top-performing using LIME**

Figure 13 illustrates how the LIME method is applied to explain the predictions of a trained model for X-ray image analysis, specifically in detecting prohibited items like knives. LIME provides interpretability by highlighting the regions of the input image that most influenced the model's decision. The three subfigures (a), (b), and (c) represent different stages of the explanation process.

In subfigure (a), the original X-ray scan of the bag is shown. As in previous figures, the knives and other dense objects are clearly visible in darker shades. This image serves as the baseline input provided to both the classifier and the LIME explanation method. In subfigure (b), LIME's superpixel segmentation is applied to the X-ray image. The yellow outlines indicate the division of the image into superpixels, which are clusters of adjacent pixels grouped based on similarity. Instead of analyzing individual pixels, LIME perturbs these superpixels to evaluate their influence on the prediction. This step is important because it reduces complexity and allows for human-interpretable regions that correspond to meaningful parts of the image, such as the blade or handle of a knife. In subfigure (c), LIME highlights the most influential superpixels for the model's prediction. The black areas represent regions that are masked out as less relevant, while the visible segments correspond to the regions that strongly contributed to the model identifying the

presence of knives. This visualization demonstrates that the classifier is focusing on the correct regions of the image (i.e., the knives), thereby increasing trust in the model's decision-making process.

Figure 13 shows how LIME enhances transparency in X-ray image classification. By decomposing the model's decision into interpretable superpixel regions, it confirms that the model is attending to the critical objects (knives) rather than irrelevant parts of the image. This approach bridges the gap between black-box AI models and human interpretability, which is essential for security-critical applications like automated baggage screening.

## 5- Conclusion

This study has presented a comprehensive framework for automated baggage threat detection, integrating the YOLOv8 deep learning model, genetic algorithm-based hyperparameter optimization, and explainable AI techniques such as Grad-CAM and LIME. The results obtained from extensive experimentation on the SIXray dataset demonstrate that the proposed approach is capable of achieving robust detection performance, with an overall precision of 90.5%, recall of 83.3%, mAP50 of 91.3%, and mAP50–95 of 67%. These findings highlight the effectiveness of the model in recognizing complex and overlapping prohibited objects, thereby improving the reliability of baggage screening systems. The optimization of hyperparameters through a genetic algorithm played a pivotal role in improving the model's adaptability and detection accuracy. Traditional manual tuning or grid search approaches are often constrained by high computational costs and suboptimal convergence. In contrast, the evolutionary optimization approach efficiently explored the search space, leading to the identification of effective configurations for learning rates, momentum, weight decay, and augmentation parameters. This not only enhanced the generalization capability of YOLOv8 but also contributed to the model's stability during training. The results confirm that metaheuristic-based optimization provides a more systematic and efficient pathway toward fine-tuning modern deep learning architectures in complex security applications.

In conclusion, this research contributes to the growing field of AI-powered security screening by demonstrating the synergy between state-of-the-art object detection models, advanced hyperparameter optimization, and explainable AI. The proposed system not only advances detection performance but also fosters transparency, trust, and operational feasibility. The findings underscore the transformative potential of deep learning and metaheuristics in enhancing transportation security while laying the groundwork for future research in safety-critical domains that demand accuracy, efficiency, and explainability.

## 6- Declarations

### *6-1- Author Contributions*

Conceptualization, A.M. and C.A.R.; methodology, A.M.; software, A.M.; validation, A.M., M.H. A.F., and C.A.R.; formal analysis, A.M. and M.H.; investigation, A.M. and C.A.R.; resources, A.M.; data curation, C.A.R.; writing—original draft preparation, A.M.; writing—review and editing, A.M., M.H., A.F., and C.A.R.; visualization, A.M.; supervision, M.H., A.F., and C.A.R.; project administration, C.A.R.; funding acquisition, C.A.R. All authors have read and agreed to the published version of the manuscript.

### *6-2- Data Availability Statement*

Data available in a publicly accessible repository: Available online: https://github.com/MeioJane/SIXray (accessed on January 2026).

### *6-3- Funding and Acknowledgments*

This research project is supported by Second Century Fund (C2F), Chulalongkorn University, Thailand. We gratefully appreciate this support.

### *6-4- Institutional Review Board Statement*

Not applicable.

### *6-5- Informed Consent Statement*

Not applicable.

### *6-6- Conflicts of Interest*

The authors declare that there is no conflict of interest regarding the publication of this manuscript. In addition, the ethical issues, including plagiarism, informed consent, misconduct, data fabrication and/or falsification, double publication and/or submission, and redundancies have been completely observed by the authors.

## 7- References


[1] Travel off Path (2025). These Are The 10 Most Visited Cities In The World Right Now. Travel off Path, Glenville, United States. Available online: https://www.traveloffpath.com/these-are-the-10-most-visited-cities-in-the-world-right-now/ (accessed on January 2026).

[2] Seyfi, G., Esme, E., Yilmaz, M., & Kiran, M. S. (2024). A literature review on deep learning algorithms for analysis of X-ray images. International Journal of Machine Learning and Cybernetics, 15(4), 1165–1181. doi:10.1007/s13042-023-01961-z.

[3] Ayesha, J. A. K., Ahmad, W., Nadeem, M., Zahra, S. W., Arshad, A., Riaz, S., & Shahid, U. (2023). Baggage Detection and Recognition Using Local Tri-Directional Pattern. International Journal of Mobile Computing Technology, 1(1), 8-17.

[4] Wang, B., Tian, Y., Wang, J., Hu, J., Liu, D., & Xu, Z. (2023). Detect occluded items in X-ray baggage inspection. Computers and Graphics (Pergamon), 115, 148–157. doi:10.1016/j.cag.2023.07.013.

[5] Ahmed, A., Velayudhan, D., Hassan, T., Bennamoun, M., Damiani, E., & Werghi, N. (2024). Enhancing security in X-ray baggage scans: A contour-driven learning approach for abnormality classification and instance segmentation. Engineering Applications of Artificial Intelligence, 130, 107639. doi:10.1016/j.engappai.2023.107639.

[6] Otabir, S. A. G., Tiang, S. S., Lim, W. H., Leong, H. Y., & Sun, B. (2023). X-Ray Baggage Object Detection Using Neural Networks Approach for Safety Purpose. Lecture Notes in Electrical Engineering, 988, 341–350. doi:10.1007/978-981-19-8703-8_30.

[7] Migel, S., Zaliskyi, M., Zavhorodnii, S., & Osipchuk, A. (2023). A New Method of Handguns Recognition While Inspecting Baggage for Aviation Security Service. Lecture Notes in Networks and Systems: Vol. 736 LNNS, 194–205. doi:10.1007/978-3-031-38082-2_15.

[8] Hassan, T., Akcay, S., Hassan, B., Bennamoun, M., Khan, S., Dias, J., & Werghi, N. (2023). Cascaded structure tensor for robust baggage threat detection. Neural Computing and Applications, 35(15), 11269–11285. doi:10.1007/s00521-023-08296-4.

[9] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2016-December, 779–788. doi:10.1109/CVPR.2016.91.

[10] Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 580–587. doi:10.1109/CVPR.2014.81.

[11] Isa, I. S., Rosli, M. S. A., Yusof, U. K., Maruzuki, M. I. F., & Sulaiman, S. N. (2022). Optimizing the Hyperparameter Tuning of YOLOv5 for Underwater Detection. IEEE Access, 10, 52818–52831. doi:10.1109/ACCESS.2022.3174583.

[12] Karaman, A., Pacal, I., Basturk, A., Akay, B., Nalbantoglu, U., Coskun, S., Sahin, O., & Karaboga, D. (2023). Robust real-time polyp detection system design based on YOLO algorithms by optimizing activation functions and hyper-parameters with artificial bee colony (ABC). Expert Systems with Applications, 221, 119741. doi:10.1016/j.eswa.2023.119741.

[13] Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). "Why should I trust you?" Explaining the predictions of any classifier. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 1135-1144. doi:10.1145/2939672.2939778.

[14] Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., & Batra, D. (2020). Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization. International Journal of Computer Vision, 128(2), 336–359. doi:10.1007/s11263-019-01228-7.

[15] Bhowmik, N., Gaus, Y. F. A., Akcay, S., Barker, J. W., & Breckon, T. P. (2019). On the impact of object and sub-component level segmentation strategies for supervised anomaly detection within X-ray security imagery. Proceedings - 18th IEEE International Conference on Machine Learning and Applications, ICMLA 2019, 986–991. doi:10.1109/ICMLA.2019.00168.

[16] Gaus, Y. F. A., Bhowmik, N., Akcay, S., Guillen-Garcia, P. M., Barker, J. W., & Breckon, T. P. (2019). Evaluation of a Dual Convolutional Neural Network Architecture for Object-wise Anomaly Detection in Cluttered X-ray Security Imagery. Proceedings of the International Joint Conference on Neural Networks, 2019-July, 1–8. doi:10.1109/IJCNN.2019.8851829.

[17] Gaus, Y. F. A., Bhowmik, N., Akcay, S., & Breckon, T. (2019). Evaluating the transferability and adversarial discrimination of convolutional neural networks for threat object detection and classification within x-ray security imagery. Proceedings - 18th IEEE International Conference on Machine Learning and Applications, ICMLA 2019, 420–425. doi:10.1109/ICMLA.2019.00079.

[18] Liang, K. J., Sigman, J. B., Spell, G. P., Strellis, D., Chang, W., Liu, F., ... & Carin, L. (2019). Toward automatic threat recognition for airport X-ray baggage screening with deep convolutional object detection. arXiv preprint, arXiv:1912.06329. doi:10.48550/arXiv.1912.06329.

[19] Miao, C., Xie, L., Wan, F., Su, C., Liu, H., Jiao, J., & Ye, Q. (2019). Sixray: A large-scale security inspection x-ray benchmark for prohibited item discovery in overlapping images. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2019-June, 2114–2123. doi:10.1109/CVPR.2019.00222.

[20] Chouai, M., Merah, M., & Mimi, M. (2020). CH-Net: Deep adversarial autoencoders for semantic segmentation in X-ray images of cabin baggage screening at airports. Journal of Transportation Security, 13(1–2), 71–89. doi:10.1007/s12198-020-00211-5.

[21] Wei, Y., & Liu, X. (2020). Dangerous goods detection based on transfer learning in X-ray images. Neural computing and applications, 32(12), 8711-8724. doi:10.1007/s00521-019-04360-0.

[22] Yao, S. Q., Su, Z. G., Yang, J. F., & Zhang, H. G. (2021). A prohibited items identification approach based on semantic segmentation. Optoelectronics Letters, 17(4), 247-251. doi:10.1007/s11801-021-0017-6.

[23] Dumagpi, J. K., & Jeong, Y. J. (2021). Pixel-level analysis for enhancing threat detection in large-scale x-ray security images. Applied Sciences (Switzerland), 11(21), 10261. doi:10.3390/app112110261.

[24] Ma, B., Jia, T., Su, M., Jia, X., Chen, D., & Zhang, Y. (2023). Automated Segmentation of Prohibited Items in X-Ray Baggage Images Using Dense De-Overlap Attention Snake. IEEE Transactions on Multimedia, 25, 4374–4386. doi:10.1109/TMM.2022.3174339.

[25] Chang, A., Zhang, Y., Zhang, S., Zhong, L., & Zhang, L. (2022). Detecting prohibited objects with physical size constraint from cluttered X-ray baggage images. Knowledge-Based Systems, 237, 107916. doi:10.1016/j.knosys.2021.107916.

[26] Fang, C., Liu, J., Han, P., Chen, M., & Liao, D. (2023). FSVM: A Few-Shot Threat Detection Method for X-ray Security Images. Sensors, 23(8), 4069. doi:10.3390/s23084069.

[27] Wei, Q., Ma, S., Tang, S., Li, B., Shen, J., Xu, Y., & Fan, J. (2023). A deep learning-based recognition for dangerous objects imaged in X-ray security inspection device. Journal of X-Ray Science and Technology, 31(1), 13–26. doi:10.3233/XST-221210.

[28] Andriyanov, N. (2024). Using ArcFace Loss Function and Softmax with Temperature Activation Function for Improvement in X-ray Baggage Image Classification Quality. Mathematics, 12(16), 2547. doi:10.3390/math12162547.

[29] Belal, M., Ahmed, A., Velayudhan, D., Hassan, T., Damiani, E., & Werghi, N. (2024). Addressing Class Imbalance in X-ray Threat Detection with Self-Supervised Balanced DINO. International Conference on Engineering and Emerging Technologies, ICEET, 2024, 1–6. doi:10.1109/ICEET65156.2024.10913663.

[30] Khan, S. M., Usman Akram, M., Salam, A. A., & Nasim, A. (2025). A Robust Semantic Segmentation Framework for Baggage Threat Detection. 2nd International Conference on Emerging Technologies in Electronics, Computing and Communication, ICETECC 2025, 1–6. doi:10.1109/ICETECC65365.2025.11071247.

[31] Nasim, A., Mazhar Khan, S., Abdul Salam, A., Shaukat, A., Hassan, T., Syed, A. M., & Usman Akram, M. (2025). Class and Data-Incremental Learning Framework for Baggage Threat Segmentation via Knowledge Distillation. IEEE Access, 13, 3574919. doi:10.1109/ACCESS.2025.3574919.

[32] Jiang, P., Ergu, D., Liu, F., Cai, Y., & Ma, B. (2021). A Review of Yolo Algorithm Developments. Procedia Computer Science, 199, 1066–1073. doi:10.1016/j.procs.2022.01.135.

[33] Dumitriu, A., Tatui, F., Miron, F., Ionescu, R. T., & Timofte, R. (2023). Rip Current Segmentation: A Novel Benchmark and YOLOv8 Baseline Results. IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, 2023-June, 1261–1271. doi:10.1109/CVPRW59228.2023.00133.

[34] Lou, H., Duan, X., Guo, J., Liu, H., Gu, J., Bi, L., & Chen, H. (2023). DC-YOLOv8: Small-Size Object Detection Algorithm Based on Camera Sensor. Electronics (Switzerland), 12(10), 2323. doi:10.3390/electronics12102323.

[35] Terven, J., Córdova-Esparza, D. M., & Romero-González, J. A. (2023). A Comprehensive Review of YOLO Architectures in Computer Vision: From YOLOv1 to YOLOv8 and YOLO-NAS. Machine Learning and Knowledge Extraction, 5(4), 1680–1716. doi:10.3390/make5040083.

[36] Jocher, G., Chaurasia, A., Qiu, J. (2023). YOLO by Ultralytics. Available online: https://github.com/ultralytics/ultralytics (accessed on December 2025).

[37] Rath, S. (2023). YOLOv8: Comprehensive Guide to State of the Art Object Detection. Available online: https://learnopencv.com/ultralytics-yolov8/#YOLOv8-vs-%20YOLOv5 (accessed on September 2025).

[38] Talaat, F. M., & ZainEldin, H. (2023). An improved fire detection approach based on YOLO-v8 for smart cities. Neural Computing and Applications, 35(28), 20939–20954. doi:10.1007/s00521-023-08809-1.

[39] RangeKing. (2023). Brief summary of YOLOv8 model structure #189. GitHub. Available online: https://github.com/ultralytics/ultralytics/issues/189 (accessed on December 2025).

[40] Li, Y., Fan, Q., Huang, H., Han, Z., & Gu, Q. (2023). A Modified YOLOv8 Detection Network for UAV Aerial Image Recognition. Drones, 7(5), 304. doi:10.3390/drones7050304.

[41] He, K., Zhang, X., Ren, S., & Sun, J. (2015). Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, 37(9), 1904–1916. doi:10.1109/TPAMI.2015.2389824.

[42] Wang, X., Gao, H., Jia, Z., & Li, Z. (2023). BL-YOLOv8: An improved road defect detection model based on YOLOv8. Sensors, 23(20), 8361. doi:10.3390/s23208361.

[43] Lin, T. Y., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). Feature pyramid networks for object detection. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2117-2125.

[44] Liu, S., Qi, L., Qin, H., Shi, J., & Jia, J. (2018). Path Aggregation Network for Instance Segmentation. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 8759–8768. doi:10.1109/CVPR.2018.00913.

[45] Zheng, Z., Wang, P., Liu, W., Li, J., Ye, R., & Ren, D. (2020). Distance-IoU loss: Faster and better learning for bounding box regression. AAAI 2020 - 34th AAAI Conference on Artificial Intelligence, 34(07), 12993–13000. doi:10.1609/aaai.v34i07.6999.

[46] Li, X., Wang, W., Wu, L., Chen, S., Hu, X., Li, J., Tang, J., & Yang, J. (2020). Generalized focal loss: Learning qualified and distributed bounding boxes for dense object detection. Advances in Neural Information Processing Systems, 2020-December, 21002–21012.

[47] Yang, L., & Shami, A. (2020). On hyperparameter optimization of machine learning algorithms: Theory and practice. Neurocomputing, 415, 295–316. doi:10.1016/j.neucom.2020.07.061.

[48] Bischl, B., Binder, M., Lang, M., Pielok, T., Richter, J., Coors, S., Thomas, T., Becker, M., Boulesteix, A. L., Deng, D., & Lindauer, M. Hyperparameter optimization: Foundations, algorithms, best practices, and open challenges. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 13(2), 1484.

[49] Decastro-García, N., Muñoz Castañeda, Á. L., Escudero García, D., & Carriegos, M. V. (2019). Effect of the Sampling of a Dataset in the Hyperparameter Optimization Phase over the Efficiency of a Machine Learning Algorithm. Complexity, 2019(1), 6278908. doi:10.1155/2019/6278908.

[50] Xu, L., Yan, W., & Ji, J. (2023). The research of a novel WOG-YOLO algorithm for autonomous driving object detection. Scientific Reports, 13(1), 3699. doi:10.1038/s41598-023-30409-1.

[51] Abreu, S. (2019). Automated architecture design for deep neural networks. arXiv preprint, arXiv:1908.10714. doi:10.48550/arXiv.1908.10714.

[52] Junior, F. A., & Suharjito. (2023). Video based oil palm ripeness detection model using deep learning. Heliyon, 9(1), e13036. doi:10.1016/j.heliyon.2023.e13036.

[53] Ali, Y. A., Awwad, E. M., Al-Razgan, M., & Maarouf, A. (2023). Hyperparameter search for machine learning algorithms for optimizing the computational complexity. Processes, 11(2), 349. doi:10.3390/pr11020349.

[54] Snustad, D. P., & Simmons, M. J. (2015). Principles of genetics. John Wiley & Sons, New Jersey, United States.

[55] Pierce, B. A. (2012). Genetics essentials: concepts and connections. WH Freeman, New York, United States.

[56] Slowik, A., & Kwasnicka, H. (2020). Evolutionary algorithms and their applications to engineering problems. Neural Computing and Applications, 32(16), 12363–12379. doi:10.1007/s00521-020-04832-8.

[57] Bäck, T., & Schwefel, H.-P. (1993). An Overview of Evolutionary Algorithms for Parameter Optimization. Evolutionary Computation, 1(1), 1–23. doi:10.1162/evco.1993.1.1.1.

[58] Simonyan, K., Vedaldi, A., & Zisserman, A. (2014). Deep inside convolutional networks: Visualising image classification models and saliency maps. 2nd International Conference on Learning Representations, ICLR 2014 - Workshop Track Proceedings, 1-8.

[59] Koh, P. W., & Liang, P. (2017). Understanding black-box predictions via influence functions. 34th International Conference on Machine Learning, ICML 2017, 4, 2976–2987.

[60] Zeiler, M. D., & Fergus, R. (2014). Visualizing and understanding convolutional networks. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 8689 LNCS(Part 1), 818–833. doi:10.1007/978-3-319-10590-1_53.

# Appendix I

**Table A.1. Summary of the acronyms used**

| Acronyms | Descriptions | Acronyms | Descriptions |
|---|---|---|---|
| ADM | Attention Deforming Module | AP | Average Precision |
| CIoU | Complete Intersection over Union | CNN | Convolutional Neural Networks |
| CUDA | Compute Unified Device Architecture | DAD | Durham Adversarial Dataset |
| Dbf3 | Durham Dataset Full Three-class | DDoAS | Dense De-overlap Attention Snake |
| DDoM | Dense De-overlap Module | DeconvNet | Deconvolution Network |
| DFL | Distribution Focal Loss | ELAN | Efficient Layer Aggregation Network |
| FCN | Fully Convolutional Networks | F-RCNN | Fast Region-based Convolutional Neural Networks |
| FSOD | Few Shot Object Detection | FSVM | Few Shot Support Vector Machine |
| FPN | Feature Pyramid Networks | GDDR6 | Graphics Double Data Rate 6 |
| GDXray | Grima X-ray Dataset | Grad-CAM | Gradient-weighted Class Activation Mapping |
| GPU | Graphical Processing Unit | HDTS | Hi-Tech Detection Systems Society |
| IoU | Intersection over Union | LIME | Local Interpretable Model-Agnostic Explanations |
| mAP | Mean Average Precision | mAP50 | Mean Average Precision calculated at an intersection over union (IoU) threshold of 0.50 |
| mAP50-95 | Mean Average Precision at IoU thresholds from 0.5 to 0.95 | O2OFM | One-to-One Fusion Module |
| P | Precision | R | Recall |
| RCNN | Region-based Convolutional Neural Networks | ResNet | Residual Network |
| SAM | Spatial Attention Module | SPP | Spatial Pyramid Pooling |
| SPPF | Spatial Pyramid Pooling - Fast | SSD | Single Shot Detection |
| SVM | Support Vector Machine | TPU | Tensor Processing Unit |
| TSA | Transportation Security Administration | XAI | Explainable Artificial Intelligence |
| YOLO | You Only Look Once | YOLOv3 | You Only Look Once version 3 |
| YOLOv5 | You Only Look Once version 5 | YOLOv7 | You Only Look Once version 7 |
| YOLOv8 | You Only Look Once version 8 | 620DV | 620 Dual View |