ISSN 1330-3651 (Print), ISSN 1848-6339 (Online)
Received: 14 January 2025; Accepted: 29 November 2025

https://doi.org/10.17559/TV-20250114002266
Original scientific paper

Prohibited Object Detection Utilizing Faster R-CNN in X-RAY Images

Raşit KÖKER*, Özcan SORGUN, Mustafa Çağrı KUTLU, Mehmet DEMİR

Abstract: Security checks play a significant role in reducing risks in various areas such as bus terminals, train stations, and airports. At the heart of these checks are X-ray machines, which play a crucial role in detecting weapons, knives, and other prohibited items. However, the increasing volume of goods being transported poses a significant problem: reliance on manual inspection of X-ray security images is fraught with human-induced weaknesses such as negligence, fatigue, and environmental distractions, leading to potential security breaches. This paper proposes an automated solution to address these issues by using the Faster R-CNN deep learning algorithm for detecting prohibited objects in X-ray images. The study utilises the SIXray dataset, comprising X-ray images of prohibited objects across six different categories obtained from various metro stations. The research utilises deep learning-based object detection models to identify five different types of prohibited items using Faster RCNN models (ResNet50, ResNet101, ResNet152, and Inception ResNet v2). These models were trained through transfer learning. The aim is to determine the most efficient model in terms of speed and accuracy for classifying and detecting prohibited objects in X-ray images. The evaluation revealed that the Faster R-CNN ResNet101 model, with a success rate of 90.6% at 0.5 IoU, is superior in terms of accurate classification and object location determination, as evidenced by its precision and recall metrics, mAP values, and image analyses. This comparison with YOLOv technologies revealed that the Faster R-CNN Resnet101 model is superior. This study highlights the potential of automated systems to improve security protocols, reduce costs, and speed up processing times in X-ray security screening.

Keywords: computer vision; deep learning; R-CNN; security application

1 INTRODUCTION

Image processing has become increasingly integral in various fields alongside technological advancements, ranging from simple to complex applications. It is notably prevalent in security domains, such as facial recognition and motion detection, and is widely used in vehicle license plate recognition systems [1, 2].

With its growing use and evolving applications, image processing continues to be essential in diverse fields including defect identification in production lines, disease detection in medicine, efficiency enhancement in agriculture and livestock, and reconnaissance and attack prevention in defence and aviation. In the context of security, X-Ray technology has been adopted at airports, shopping malls, and other public institutions.

Traditionally reliant on manual inspection by personnel, there is a growing shift towards computerized vision through deep learning applications. While replacing manual observation might seem disadvantageous, it is expected to minimize human error and reduce vision-related issues caused by constant screen exposure. Deep learning methods, a subset of machine learning commonly referred to as artificial intelligence, are employed in these applications. The earliest known study in this field was conducted by Ivakhnenko and Lapa (1965). [3]. Deep learning gained momentum post-2005, particularly with advancements in GPU technology and cloud computing, leading to accelerated development and broader application potential. However, this rapid progress has also raised concerns about the potential risks associated with artificial intelligence [3]. Human factors like negligence, fatigue, and environmental distractions in X-ray security systems can lead to security vulnerabilities.

This study proposes a deep learning-based system to detect prohibited items in X-ray images [4], aiming to select the best model in terms of speed and accuracy for classifying and locating these items. This approach can potentially expedite the inspection process and reduce costs. The SIXray dataset, comprising X-ray images of five prohibited items excluding hammers due to insufficient image count, was used in this paper. The second part provides an overview of artificial intelligence, machine learning [5] and deep learning [6, 7] focusing on convolutional neural networks. It examines various models and their architectures, along with the evolution and characteristics of region-based convolutional neural networks. In the third part, details about the SIXray dataset, distinctions between 2 image classification and object detection models, and the training process using Google Colab are discussed. The study evaluates and compares the results of four object detection models: Faster R-CNN ResNet50, Faster RCNN ResNet101, Faster R-CNN ResNet152, and Faster R-CNN Inception ResNet v2, using transfer learning to improve model performance and save time. The final section presents the conclusions and draws future research.

This study aims to detect objects such as guns, knives, pliers, keys, and scissors using the SIXray dataset, which consists of 18 X-ray images. The Faster R-CNN algorithm will be tested and compared by incorporating ResNet50, ResNet101, ResNet152, and Inception ResNet V2 models. Specifically, it will be investigated whether transfer learning can be used with models pre-trained on the COCO dataset, whether time savings can be achieved, and whether model performance can be improved. The models will be evaluated in stages of 25000, 50000, and 100000 steps, and the results obtained in this context will be compared with YOLOv. With the implementation of these proposals, X-ray inspection, which is currently performed manually, could transition to an autonomous, unmanned process in the near future. This transition will take place within more stable and secure structures and will increase the overall reliability of inspections. As a result, the detection of dangerous objects will become more efficient and the likelihood of adverse events will be significantly reduced. This development will facilitate the easier identification of threats, thereby preventing potential dangers and providing a safer environment.

2 BACKGROUND

CNNs, foundational in modern deep learning, automate feature extraction, eliminating the need for manual input. The first CNN architecture was introduced by LeCun in 1998, evolving into the LeNet-5 model used in banking [8]. Subsequent models like AlexNet and VGG16 significantly improved image classification, leading to the development of more complex networks like GoogLeNet and ResNet. These architectures laid the groundwork for advancements in object detection, image processing, and autonomous vehicle technology. Region-Based CNNs, including R-CNN, Fast R-CNN, and Faster R-CNN, are deep learning models used in computer vision to identify and classify objects within images. R-CNN utilizes selective search for region proposals, but its reliance on these for feature extraction makes it slow. Fast R-CNN improves upon this by using a feature map from the entire image for region proposals, leading to better efficiency. Faster R-CNN introduces a Region Proposal Network (RPN), further accelerating the process and making it more effective. The original R-CNN model, developed by Girshick [9, 10] uses a selective search algorithm [11] to generate approximately 2000 region proposals (Regions of Interest - RoI) per input image.

These regions are then processed through a convolutional neural network for feature extraction, followed by classification using a Support Vector Machine (SVM) and bounding box regression for precise localization. However, R-CNN is slow in training and testing due to the need to process each region proposal through the network separately. To address R-CNN's shortcomings, Ross Girshick introduced Fast R-CNN [10]. Which inputs the entire image into a convolutional neural network to create a feature map. This approach, inspired by SPPNet (Spatial Pyramid Pooling Network), uses RoI pooling instead of maximum pooling to handle regions. The resulting feature map is then split into a SoftMax 3 classifier and a bounding box regressor. Fast R-CNN is significantly faster than R-CNN as it avoids processing each region proposal through the convolutional network repeatedly. Faster R-CNN, developed by Ren et al. [12]. eliminates the need for selective search by introducing a Region Proposal Network (RPN). This network directly generates region proposals from the feature map. The RPN guides the Fast R-CNN network on where to focus, significantly speeding up the process. Faster R-CNN has been shown to outperform both R-CNN and Fast R-CNN in terms of speed [12].To support these statements, the CNN structure is shown in Fig. 1.

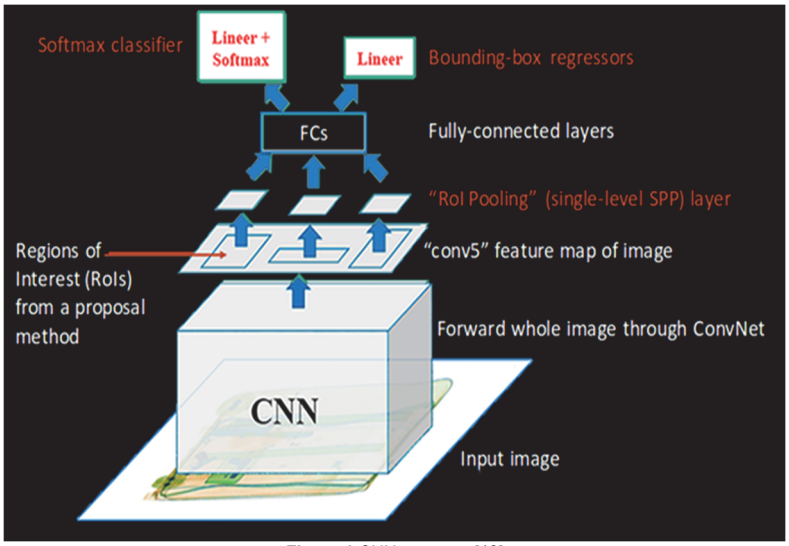

Figure 1 CNN structure [13]

In the context of X-ray image analysis for baggage security, several studies have been conducted using different datasets and CNN models, each achieving varying degrees of success in detecting [14] items like weapons and tools. The research within this paper extends these efforts by employing the SIXray dataset to classify multiple object types and comparing the results of different models. Considering the vast amount of existing literature, the paper focuses on using the SIXray dataset for classifying objects like weapons and tools in X-ray images. Various CNN models are trained and compared, considering factors such as accuracy and processing speed, to optimize object detection in security settings.

3 METHODOLOGIES

This article aims to detect prohibited objects in X-ray images using deep learning models. A comparative study was conducted between Faster R-CNN models (ResNet50, ResNet101.

ResNet152, and Inception ResNet v2) to determine the most suitable model for this task, drawing on the literature for hyperparameters. Subsequently, the best results obtained from these comparisons were compared with the YOLOv (You Only Look Once) model, which provides better predictions than SSD (Single Shot Multibox Detector) and EfficientDet, the most widely used models in today's technologies, in order to increase generalisation. In this context, the SIXray dataset, which contains approximately 1 million positive and negative examples collected from several metro stations in real environments, was used [15, 16]. This dataset is publicly available open-source data. This dataset contains objects commonly prohibited in six categories: Guns, Knives, Wrenches, Pliers, Scissors, and Hammers. Due to the imbalanced structure of the dataset, the significant excess of negative

examples, the presence of unclear and dark images, and the very low representation of the Hammer category, it was excluded from this study. The curated data is shown in Fig. 2.

Figure 2 Content of the dataset [15]

Object Detection Unlike traditional image classification, object detection models not only classify objects but also locate them within the image with bounding boxes. This approach was chosen due to the multiple object types 4 and their varying sizes and positions in the images, making classification models unsuitable for this task. The steps to be followed in order to trans the model and then evaluate it are shown in Fig. 3.

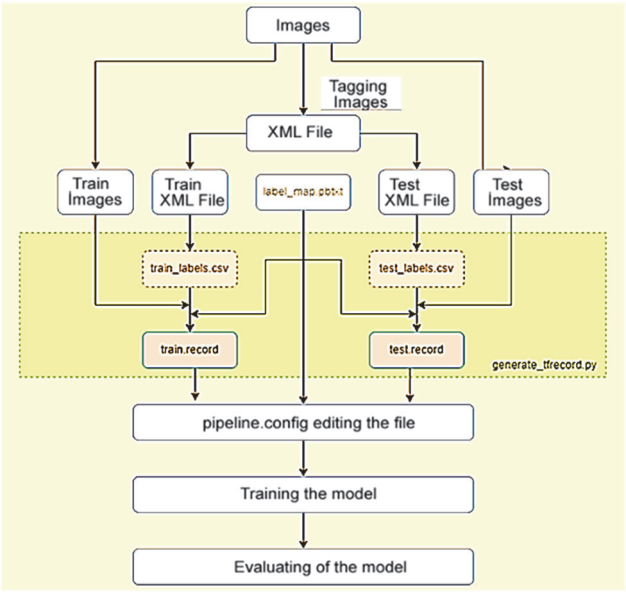

Figure 3 The steps to be followed in order to train the model and then evaluate

Training Environment and Setup Model training was conducted using Google Colab with GPU acceleration, employing the TensorFlow library (version 2.8.0, February 2022). The TensorFlow Object Detection API, pretrained on the COCO dataset, was used to facilitate the creation of object detection models. Python 3.7 was used throughout the study. Training Preparations and Model Training Initial steps included setting up a workspace in Google Colab, linking to Google Drive for data persistence, and downloading the TensorFlow Model Zoo via Git. The models were trained using transfer learning from pre-trained Faster R-CNN models. For this study, a total of 8916 images containing intertwined or over lapping objects were labeled from scratch. This is shown in Tab. 1.

Table 1 Number of the images and labels in the training and test data sets.

	Number of İmages	Number of Tags				
		Gun	Knife	Wrench	Pliers	Scissors
Train	7176	4057	2496	3040	5439	1104
Test	1740	994	649	728	1333	262

The training process involved labelling images using the labelling tool, generating XML files for each image, and converting them into TFRecord format. The models were then trained in stages, initially for 25000 steps and subsequently for 50000 and 100000 steps, with adjustments made based on interim performance evaluations. Evaluation of Training Results Models were evaluated based on their performance at 50000 and 100000 training steps. Evaluation metrics included confusion matrices, precision, recall, and mean Average Precision (mAP) at an Intersection over Union (IoU) threshold of 0.5. After thorough testing and evaluation, including assessing the models' ability to detect objects in 100 test images, the Faster R-CNN ResNet101 model was selected as the most suitable model based on its balance of speed and performance.

4 RESULTS AND DISCUSSION

Within the scope of this study, what a convolutional neural network is, how it works and what kind of operations are performed on its layers were explained. It was stated that convolutional neural networks are used for feature extraction in images and can classify only a single

object in an image. Since it is necessary to classify multiple objects in the images in the data set and detect objects in different locations and sizes, instead of image classification models that only classify objects, object detection models that find the locations of objects in addition to classification were used. The architecture, features, advantages and disadvantages of object detection models were explained. Model training was carried out using GPU on Google Colab. The Tensorflow library, Tensorflow Object Detection API and many packages used in training preparations and training processes were installed. The SIXray dataset containing X-ray images was used. In the images in this data set, objects such as guns, knives, pliers, wrenches and scissors were labelled from scratch. Within the scope of the study, ResNet50, ResNet101, ResNet152 and Inception ResNet V2 models of the Faster R-CNN algorithm were used and compared. Transfer learning was performed using models previously trained on the COCO dataset provided by the Tensorflow Object Detection API. Thus, time was saved and model success was increased. After the necessary training preparations were completed, the model was trained in stages with 25000, 50000 and 10000 step numbers. Error loss changes were observed via Tensorboard throughout the training processes. At the end of each stage, the precision and sensitivity values and confusion matrix results were examined and the models were compared.

After 50000 steps, the Faster R-CNN Inception ResNet v2 model was eliminated because it fell behind in terms of performance compared to other models, and training was continued with other models. Finally, after the model was trained for 100000 steps, the Faster R-CNN ResNet101 model, which was observed to perform better classification and positioning according to the precision and mAP values and as a result of the examination of 100 images with object detection, was the best in terms of speed-performance with a success rate of 90.6% for 0.5 IoU. This was obtained as a suitable model. The models obtained within the scope of this study were trained in stages with determined number of steps. At the end of these stages, evaluation results were obtained for each model. For example, the commands required to obtain evaluation results for the ResNet101 model are shown in Fig. 4.

```
%%bash
cd "/content/drive/MyDrive/tensorflow/workspace"
python model_main_tf2.py --model_dir="my-models/faster_rcnn_resnet101_v1" \
        --pipeline_config_path="my-models/faster_rcnn_resnet101_v1/pipeline.config" \
        --checkpoint_dir="my-models/faster_rcnn_resnet101_v1"
```

Figure 4 Commands required to perform model evaluation

All models were started in training with the momentum optimization method, with a learning rate of 0.04 and a momentum value of 0.9. Since the results of the models, which were first trained for 25000 steps, did not meet the needs, the training was continued with 50000 steps. The change in classification and total error values in the models over 50000 steps is shown in Fig. 5.

The models were evaluated after being trained for 50000 steps. During the evaluation phase of the models, the confusion matrix and the accompanying calculated criteria were used to measure the success of detecting objects in the images. The confusion matrix shows the number of incorrect and correct predictions. The confusion matrix and calculations are shown in Fig. 6.

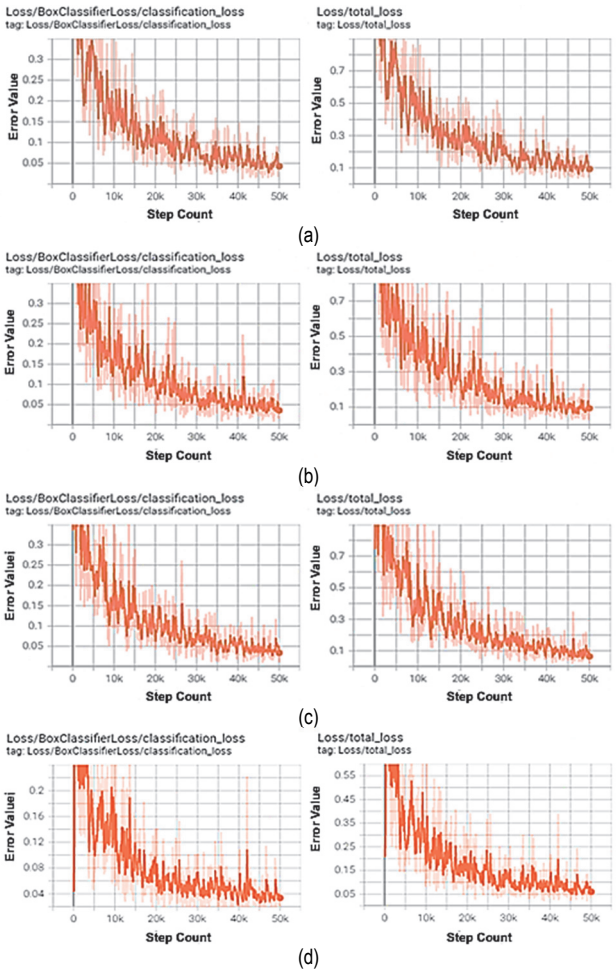

Figure 5 Error changes in the models over 50000 steps: (a) Faster R-CNN ResNet50 model, (b) Faster R-CNN ResNet101 model, (c) Faster R-CNN ResNet152 model, (d) Faster R-CNN Inception ResNet v2 model

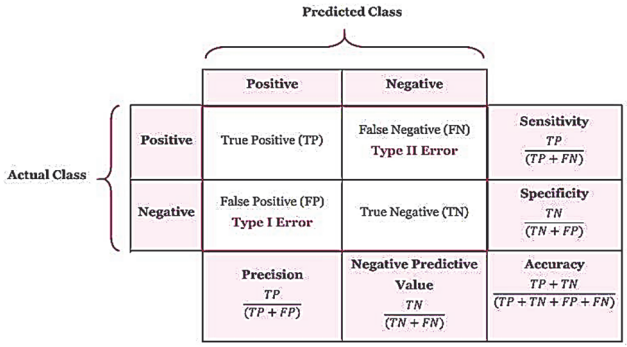

Figure 6 Confusion matrix and calculations used in the evaluation [17]

Intersection over Union (IoU) is a measure that shows how much two bounding boxes, our estimated boundary, overlap with the ground truth. The IoU value (for example, 0.5) is used to determine whether a detection is valid or not. The IoU is found by dividing the intersecting area between the predicted bounding box and the real object bounding box by the junction area between them. This situation is shown in Fig. 7. Confusion matrices of the models after 50000 steps are shown in Tab. 2, Tab. 3, Tab. 4 and Tab. 5. The values on the diagonal of the confusion matrix indicate objects that are correctly predicted (TP - true positive), the values in row 6 indicate objects that do not actually exist but are detected (FP - false positive), and the values in column 6 indicate objects that actually exist but cannot be detected (FN - false negative). For the remaining

values, the ones in the row (FN - false negative) and the ones in the column (FP - false positive) show for each class. For example; in Tab. 2, in the section where the 2nd row and the 4th column intersect, 4 objects were perceived as pliers, although they should have been perceived as knives. In the section where the 4th Row and the 1st Column intersect, 1 object was perceived as a gun, although it should have been perceived as pliers.

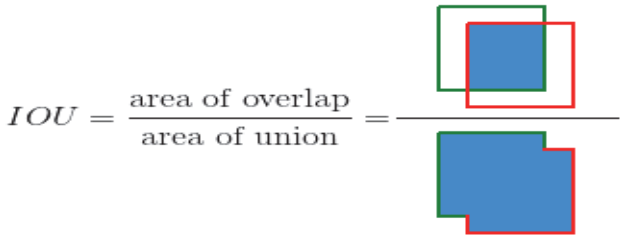

Figure 7 Intersection over union (IoU) [18, 19]

Table 2 Confusion matrix for Faster R-CNN ResNet50 model after 50000 steps

Real	Prediction					
	Gun	Knife	Wrench	Pliers	Scissors	BG
Gun	964	4	0	3	0	23
Knife	2	541	3	4	2	97
Wrench	0	2	611	24	1	90
Pliers	1	1	38	1124	3	166
Scissors	1	3	1	2	215	40
BG	40	69	156	158	14	0

Table 3 Confusion matrix for Faster R-CNN ResNet101 model after 50000 steps

Real	Prediction					
	Gun	Knife	Wrench	Pliers	Scissors	BG
Gun	955	2	1	2	0	34
Knife	1	555	2	1	2	88
Wrench	1	4	590	20	0	113
Pliers	0	4	19	1122	5	183
Scissors	1	3	0	0	219	39
BG	29	63	85	107	16	0

Table 4 Confusion matrix for Faster R-CNN ResNet152 model after 50000 steps

Real	Prediction					
	Gun	Knife	Wrench	Pliers	Scissors	BG
Gun	968	1	2	1	0	22
Knife	1	542	3	1	1	101
Wrench	1	2	589	23	0	113
Pliers	3	0	18	1123	5	184
Scissors	1	3	0	1	219	38
BG	34	43	83	87	13	0

Table 5 Confusion matrix for Faster R-CNN Inception ResNet v2 model after 50000 steps

Real	Prediction					
	Gun	Knife	Wrench	Pliers	Scissors	BG
Gun	946	0	1	1	0	46
Knife	2	478	11	1	4	153
Wrench	1	1	516	27	0	183
Pliers	2	1	49	975	2	304
Scissors	1	5	1	9	181	65
BG	44	40	155	160	28	0

Precision refers to the ratio of true positives to all positives and indicates correct predictions from predicted samples. Sensitivity (recall) gives the ratio of true positives to the total number of objects and indicates how many of the objects to be detected can be detected. Mean Precision (mAP) is the average of AP (Average Precision) values calculated for each class using the confusion matrix and provides information about the performance of the models. It takes a value between 0 and 1, and the higher it is, the more successful the model is. The precision and sensitivity values of each class for 0.5 IoU of the models after 50000 steps are given in Tab. 6, Tab. 7, Tab. 8 and Tab. 9.

Table 6 Results for Faster R-CNN ResNet50 model after 50000 steps

Class	Precision_@0.5IoU	Recall_@0.5IoU
Gun	0.956	0.970
Knife	0.873	0.834
Wrench	0.755	0.839
Pliers	0.855	0.843
Scissors	0.915	0.821

Table 7 Results for Faster R-CNN ResNet101 model after 50000 steps

Class	Precision_@0.5IoU	Recall_@0.5IoU
Gun	0.968	0.961
Knife	0.880	0.855
Wrench	0.846	0.810
Pliers	0.896	0.842
Scissors	0.905	0.836

Table 8 Results for Faster R-CNN ResNet152 model after 50000 steps

Class	Precision_@0.5IoU	Recall_@0.5IoU
Gun	0.960	0.974
Knife	0.917	0.835
Wrench	0.847	0.809
Pliers	0.909	0.842
Scissors	0.920	0.836

Table 9 Results for Faster R-CNN Inception ResNet v2 model after 50000 steps

Class	Precision_@0.5IoU	Recall_@0.5IoU
Gun	0.950	0.952
Knife	0.910	0.737
Wrench	0.704	0.709
Pliers	0.831	0.731
Scissors	0.842	0.691

Considering the low detection speed of the Inception ResNet v2 model, it lags behind other models in terms of performance in detecting wrenches, pliers and scissors. Therefore, it is not a speed-performance model. When the object detection test was performed on 100 images and the models were examined, it was seen that the number of detected objects and object positioning could be improved, and ResNet50, ResNet101 and ResNet152 models were trained up to the next stage, 100000 steps, to provide better performance. The change in classification and total error values in the models over 100000 steps is shown in Fig. 8.

Confusion matrices of the models after 100000 steps are shown in Tab. 10, Tab. 11 and Tab. 12.

Table 10 Confusion matrix for Faster R-CNN ResNet 50 model after 100000 steps

Real	Prediction					
	Gun	Knife	Wrench	Pliers	Scissors	BG
Gun	963	2	2	2	0	25
Knife	2	558	1	0	1	87
Wrench	1	2	602	25	0	98
Pliers	3	3	19	1128	1	179
Scissors	1	3	0	2	209	47
BG	37	61	86	95	7	0

Table 11 Confusion matrix for Faster RCNNResNet 101 model after 100000 steps

Real	Prediction					
	Gun	Knife	Wrench	Pliers	Scissors	BG
Gun	968	1	0	1	0	24
Knife	1	550	2	1	5	90
Wrench	1	6	593	20	1	107
Pliers	3	2	12	1123	2	191
Scissors	1	3	0	0	223	35
BG	25	41	68	79	11	0

Table 12 Confusion matrix for Faster R-CNN ResNet 152 model after 100000 steps

Real	Prediction					
	Gun	Knife	Wrench	Pliers	Scissors	BG
Gun	964	0	2	2	0	26
Knife	1	548	0	1	2	97
Wrench	2	2	601	14	0	109
Pliers	1	1	16	1127	4	184
Scissors	0	3	0	0	221	38
BG	31	59	62	75	12	0

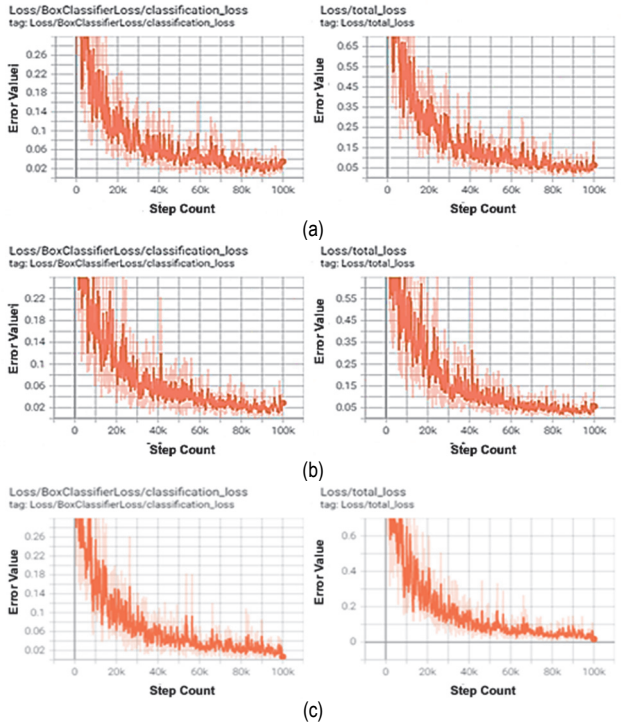

Figure 8 Error changes in the models over 100000 steps: (a) Faster R-CNN ResNet50 model, (b) Faster R-CNN ResNet101 model, (c) Faster R-CNN ResNet152 model

The precision and sensitivity values of each class for 0.5 IoU of the models after 100000 steps are given in Tab. 13, Tab. 14 and Tab. 15.

Table 13 Results for Faster R-CNN ResNet50 model after 100000 steps

Class	Precision_@0.5IoU	Recall_@0.5IoU	mAP_@0.5IoU
Gun	0.956	0.969	0.905
Knife	0.887	0.860	
Wrench	0.848	0.827	
Pliers	0.901	0.846	
Scissors	0.959	0.798	

Table 14 Results for Faster R-CNN ResNet101 model after 100000 steps

Class	Precision_@0.5IoU	Recall_@0.5IoU	mAP_@0.5IoU
Gun	0.969	0.974	0.906
Knife	0.912	0.847	
Wrench	0.879	0.815	
Pliers	0.917	0.842	
Scissors	0.921	0.851	

Table 15 Results for Faster R-CNN ResNet152 model after 100000 steps

Class	precision@0.5IoU	recall@0.5IoU	mAP@0.5IoU
Gun	0.965	0.970	0.904
Knife	0.894	0.844	
Wrench	0.883	0.826	
Pliers	0.925	0.845	
Scissors	0.925	0.844	

After 100000 steps, 6 out of 100 images tested were identified. In the images in Fig. 9, detections above 30% success are stated; but the confusion matrix groups objects according to 50% success and 0.5 IoU value. Depending on success, the detection status can optionally be set to a value such as 50%. As seen in Fig. 9, two models detect only two of the three pliers. Although the Faster R-CNN ResNet101 model detected the third pliers, the object will be considered as not detected in terms of the confusion matrix due to its 48% success rate.

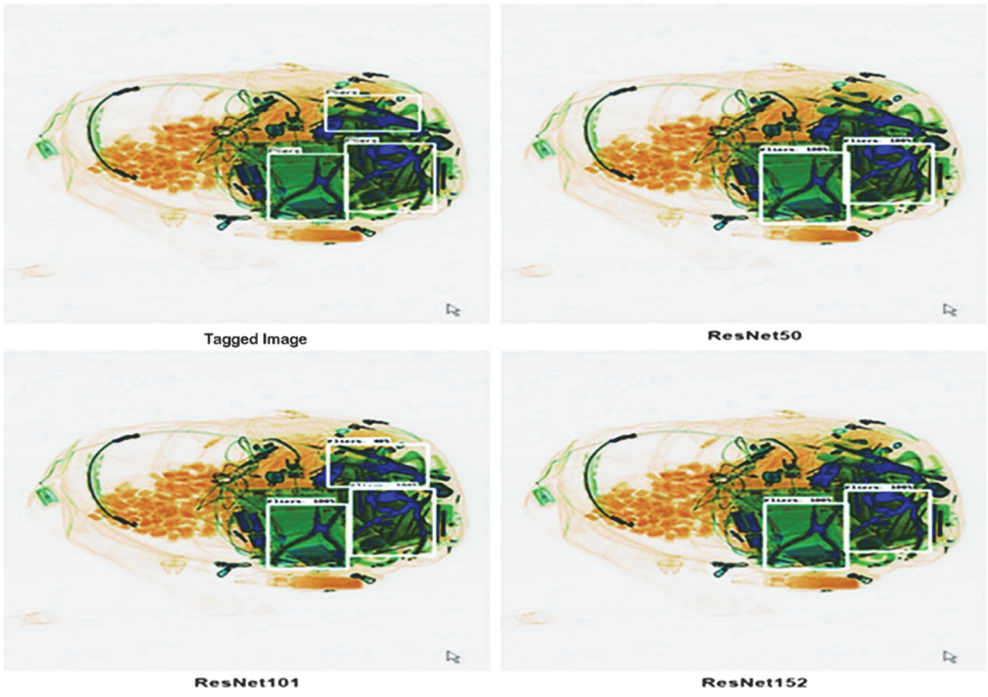

Figure 9 First image tested with object detection models

Fig. 10 shows that although the pliers are detected by all three models, only the Faster R-CNN ResNet50 and Faster R-CNN ResNet101 models detect the wrench at 99% and 100%, respectively.

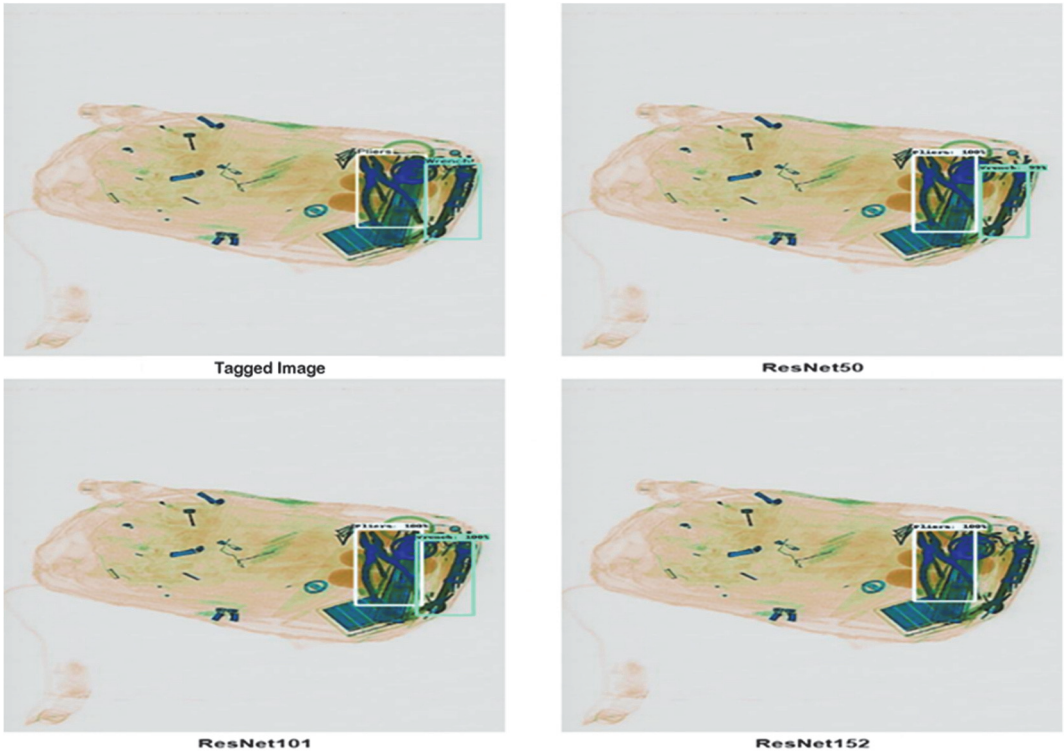

Figure 10 Second image tested with object detection models

In the image in Fig. 11, there are three wrenches and two pliers. All three models detected the pliers accurately. Only the Faster R-CNN ResNet101 model was able to detect a wrench with 83% success, while no other model could detect the other two wrenches. The Faster R-CNN ResNet50 model detects pliers instead of a wrench with a 35% success rate. However, since the success rate is below 50%, it will be ignored by the confusion matrix.

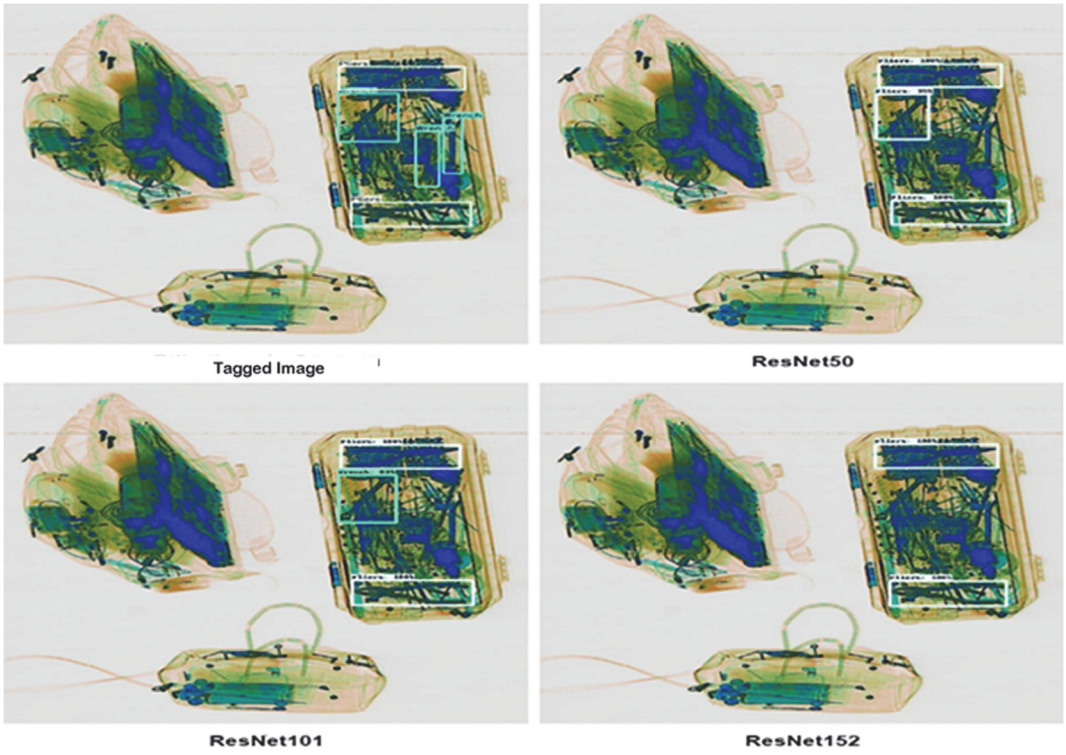

Figure 11 Third image tested with object detection models

In the image in Fig. 12, although the number of objects to be detected or not detected is high and they are nested, all objects are successfully detected by all models. Additionally, only the Faster R-CNN ResNet101 model stands out in terms of positioning.

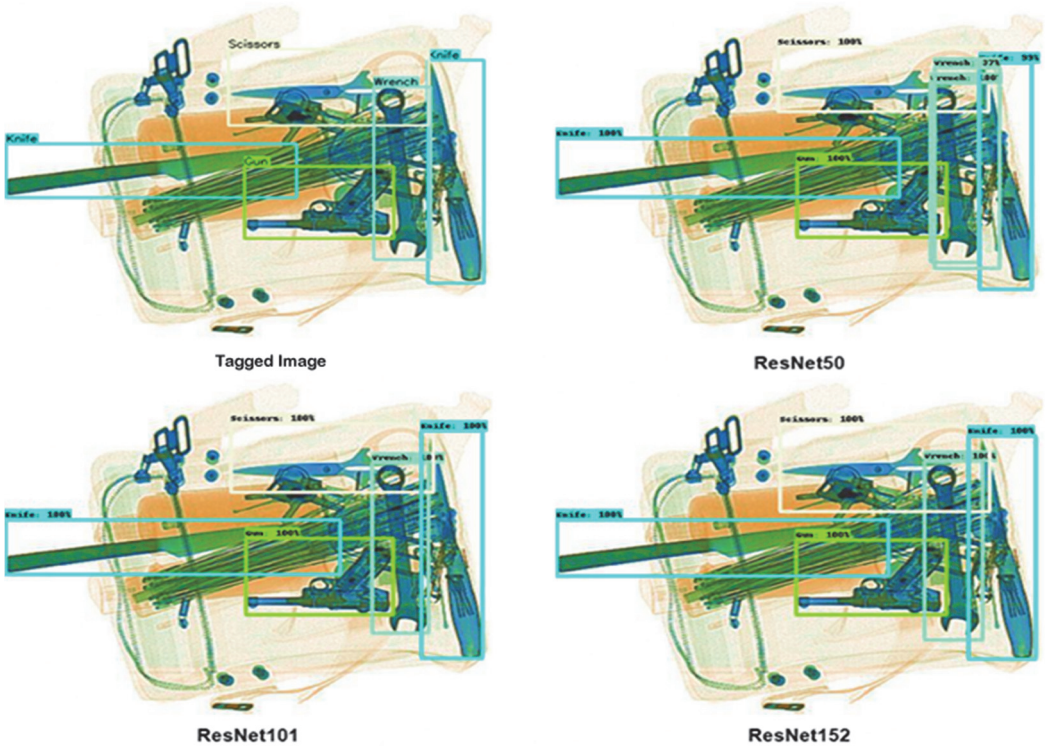

Figure 12 Fourth image tested with object detection models

It can be seen in Fig. 13 that three wrenches and two pliers were successfully detected by all three models. Only the Faster R-CNN ResNet50 model detects missing pliers with a success rate of 68%.

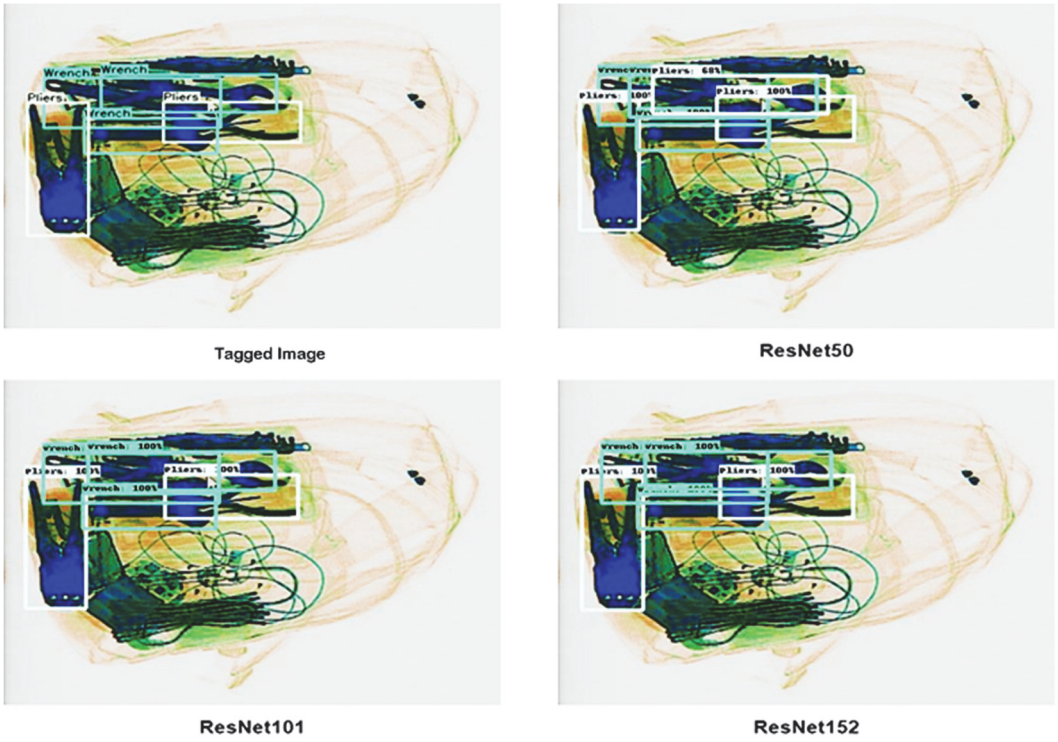

Figure 13 Fifth image tested with object detection models

As seen in Fig. 14, all objects were successfully detected by all models. Only in one of the scissors, the Faster R-CNN ResNet50 model achieved 71% success, the Faster R-CNN ResNet101 model achieved 96% success, and the Faster R-CNN ResNet152 model achieved 58% success.

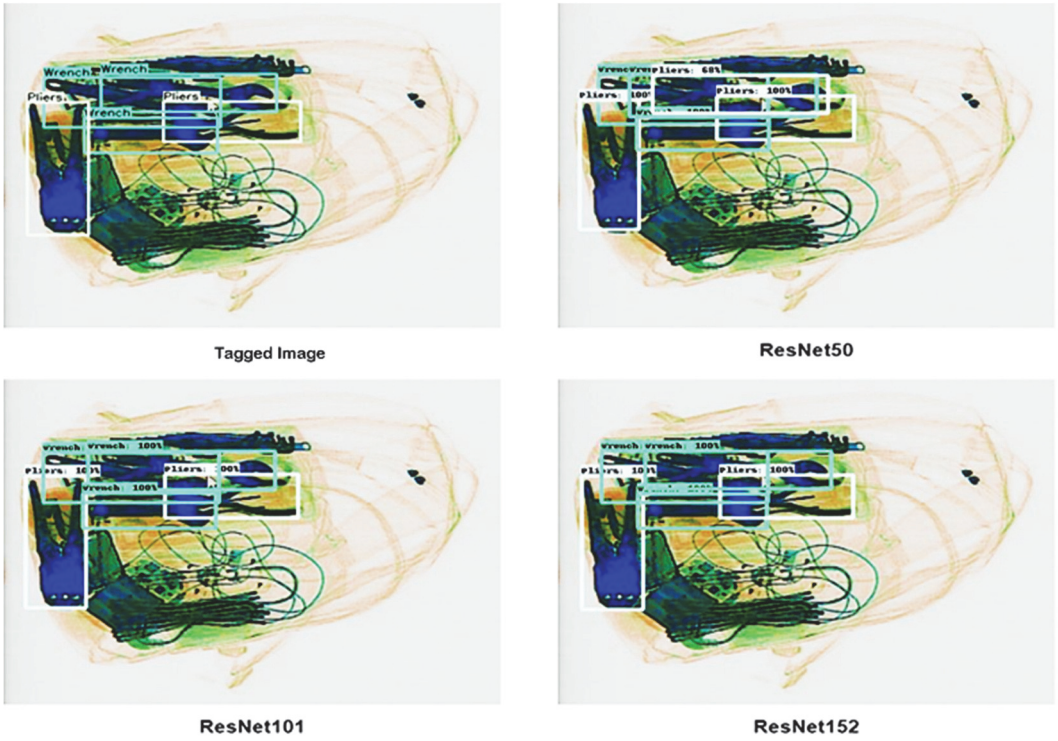

Figure 14 Sixth image tested with object detection models

When the object detection test was performed on 100 images and the models were examined, it was observed that the Faster R-CNN ResNet50 model showed very good performance in light and simple images, but its success decreased in complex and dark images. In the Faster R-CNN ResNet101 and Faster R-CNN ResNet152 models, out of 100 images, Faster R-CNN ResNet101 comes to the fore in some images, while Faster R-CNN ResNet152 comes to the fore in some images. However, when the images are examined, it is observed that in cases where Faster R-CNN ResNet101 and Faster R-CNN ResNet152 models make similar detection, the Faster R-CNN ResNet101 model performs better classification and positioning in the majority. Considering the precision and mAP values, and considering the image processing speed between Faster R-CNN ResNet101 and Faster R-CNN ResNet152, the Faster R-CNN ResNet101 model was chosen as the most suitable model in terms of speed-performance. Within the scope of this study, the Faster RCNN ResNet101 model was determined as the most suitable model. It has been determined that the selection of hyperparameters and training levels is appropriate based on the performance of the ResNet101 model in the Faster RCNN algorithm. When this selection is justified, it is understood that it achieves appropriate optimisation from the high accuracy and clear location information obtained. To increase generalisability and determine error analysis for comparison purposes, alternative algorithms currently in active use, such as YOLOv, SSD, and EfficientDet, were examined and compared with each other. The examination of these algorithms revealed that the YOLOv algorithm is efficient in terms of location, speed, and accuracy, as shown in Tab. 16.

As shown in Tab. 16, the comparison of the given algorithms determined that YOLOv8 and higher models are the most suitable. To evaluate the generalisation ability, tests were conducted on real-world X-ray datasets, and model training was performed on YOLOv9 while keeping all algorithm hyperparameters the same. The model prediction was performed using real-world dataset tests. The obtained prediction output is presented in Fig. 15, and the Faster RCNN ResNet101 output is shown in Fig. 16.

Table 16 Model Comparison.

Feature	YOLOv	SSD	EfficientDet
Speed (FPS)	Fastest	Very Fast	Fast
Accuracy (mAP)	Very High	High	High (Best Parameter Efficiency)
Architecture	Anchor-Free (YOLOv8+)	Anchor-Based	BiFPN (Efficiency Focused)
Best For	Real-Time Applications	Good Balance	Resource-Constrained/Edge Devices

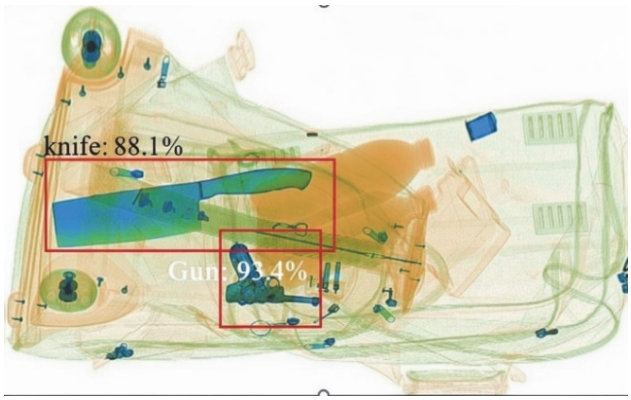

Figure 15 YOLOv9 target image

As shown in Fig. 16, it can be understood from the given visual that YOLOv9 knife can predict at a rate of 88.1%, while gun can predict at a rate of 93%. YOLOv9 determined the location information using its own method. The Faster RCNN algorithm's ResNet101 model was

determined to be 100% accurate in predicting the location of knives and 100% accurate in predicting the location of guns. This comparison shows that the Faster RCNN algorithm's ResNet101 model is more successful.

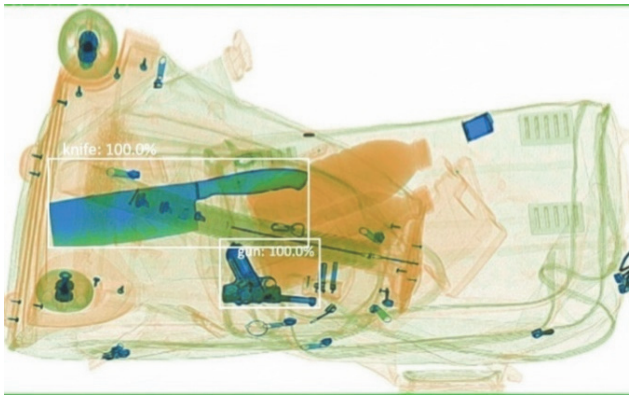

Figure 16 Faster RCNN ResNet101 target image

In order to make the study more efficient, it is understood that firstly collecting images from different object classes and improving the images by pre-processing the collected images will further improve the percentages of the model results. By collecting different object classes and improving the images and applying the most appropriate model, it is expected that the detectability of dangerous objects, including their locations, will be quite high. It is understood that different deep learning models that have been successful in terms of speed and efficiency can be used to detect harmful objects. It is thought that detecting these harmful objects will play an important role in ensuring the safety of life and property by detecting all kinds of objects that cannot be detected by the human eye in public areas, preventing negative situations. In addition, it is thought that this information will be stored in a meaningful way and used with advanced technologies that will be developed in the future.

5 CONCLUSION

Many people travel by train, bus or plane and security checks are vital to ensure safety during these journeys. Among these, baggage screening stands out as a vital security measure. This paper addresses the need for a system that can speed up security checks and improve their efficiency. The paper starts by describing convolutional neural networks (CNNs), their functions and the types of operations performed in their layers. It is emphasized that although CNNs are adept at feature extraction in images, they are limited to classifying only a single object per image. As a result, for a dataset containing images with multiple objects of different sizes and locations, object detection models were used instead of image classification models alone.

The training was performed on Google Colab using GPU, where the TensorFlow library and the TensorFlow Object Detection API played an important role. The SIXray dataset of 18 X-ray images was used and objects such as guns, knives, pliers, keys and scissors were labelled from the beginning. The study tested and compared the Faster R-CNN algorithm's ResNet50, ResNet101, ResNet152 and Inception ResNet V2 models. By using transfer learning from pre-trained models on the COCO dataset, time was saved and model performance was improved. The models were trained in stages of 25000, 50000 and 100000 steps. Throughout these steps, error losses were monitored using Tensorboard and the models were evaluated based on precision, recall and confusion matrix results. After 50000 steps, the Faster R-CNN Inception ResNet v2 model was eliminated due to its poor performance. Finally, after 100000 steps of training and analyzing the precision and mAP values and testing on 100 images for object detection, the Faster R-CNN ResNet101 model emerged as the optimal choice with 90.6% accuracy at 0.5 IoU, offering an optimal balance between speed and performance. Additionally, to increase generalisability, the model was trained using the YOLOv9 algorithm and predictions were made on real data. The results obtained from this application showed that YOLOv9 could predict knives with an accuracy of 88.1% and weapons with an accuracy of 93%, and it was determined that YOLOv9 determined location information using its own method. In this context, it was found that the prediction output of YOLOv9, the best of the advanced YOLOv SSD, and EfficientDet proposals, was lower in performance when compared to Faster R-CNN ResNet101.

In this study, negative examples were not included in the dataset due to potential risks, limitations in development, and additional costs. The reason for not including negative examples is to improve security and ethical considerations. The aim is to establish a robust infrastructure to improve performance in future studies. In future projects, these negative examples and additional positive examples may be included. The hammer class was not used in the dataset because the limited availability of images negatively affected training; in future research, this class could be included by expanding the image database or extending the study to cover other object classes. Other deep learning models promising speed and efficiency in object detection were investigated. As a result of the research, YOLOv, SSD, and EfficientDet were analysed. The YOLOv9 algorithm was studied to generate prediction output for the system's performance and applicability. This output was found to be not as efficient as the Faster RCNN algorithm ResNet101 model. Considering these potential risks, it was understood that the study created a developable infrastructure for increasing the database and increasing positive and negative data by researching real-time generated data in addition to data sets taken from real-time applications. The most significant development was the applicability of the developed infrastructure (model) to real-time systems directly through model weights. Generalisation was observed to be increased when compared to various algorithms and models. Recommendations such as creating a hybrid model and decision support system with these compared models can be made. With these suggestions, X-ray inspection and analysis, currently performed by human labour, can be carried out autonomously and unmanned in more stable structures in the coming years. This can make the detection of dangerous objects easier and prevent adverse situations. Public security can be ensured by saving on human labour and time. In these cases, it can directly contribute to the national security and the national economy.

6 REFERENCES

[1] Kapur, J. & Baregar, A. J. (2013). Security using image processing. *International Journal of Managing Information Technology*, *5*, 13-21. https://doi.org/10.5121/ijmit.2013.5202

[2] Wójcik, W., Gromaszek, K., & Junisbekov, M. (2016). *Face recognition: Issues, methods and alternative applications*. Face Recognition: Issues, Methods and Alternative Applications. InTechOpen, London, UK. https://doi.org/10.5772/62950

[3] Ivakhnenko, A. G. & Lapa, V. G. (1965). *Cybernetic predicting devices*. CCM Information Corporation.

[4] Robotti, N. (2013). The discovery of X-ray diffraction. *Rendiconti Lincei*, *24*(S1), 7-18. https://doi.org/10.1007/s12210-012-0205-1

[5] Chollet, F. (2018). *Deep learning with Python*. MITP-Verlags GmbH & Co. KG.

[6] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436-444. https://doi.org/10.1038/nature14539

[7] *Deep Learning*. (2025). Retrieved from https://towardsdatascience.com/what-is-deep-learning-and-how-does-it-work-2ce44bb692ac

[8] LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*, 2278-2324. https://doi.org/10.1109/5.726791

[9] Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 580-587. https://doi.org/10.1109/CVPR.2014.81

[10] Girshick, R. (2015). Fast R-CNN. *2015 IEEE International Conference on Computer Vision (ICCV)*, 1440-1448. https://doi.org/10.1109/ICCV.2015.169

[11] Uijlings, J. R. R., van de Sande, K. E. A., Gevers, T., & Smeulders, A. W. M. (2013). Selective search for object recognition. *International Journal of Computer Vision*, *104*, 154-171. https://doi.org/10.1007/s11263-013-0620-5

[12] Ren, S., He, K., Girshick, R., & Sun, J. (2017). Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *39*, 1137-1149. https://doi.org/10.1109/TPAMI.2016.2577031

[13] Pulcit Sharma. (2018). Retrieved from https://www.analyticsvidhya.com/blog/2018/11/implementation-faster-r-cnn-python-object-detection/

[14] Koker, R., Oz, C., & Ferikoglu, A. (2001). Development of a vision-based object classification system for an industrial robotic manipulator. *Proceedings of the 2001 IEEE International Conference on Industrial Technology (ICIT 2001)*, 1281-1284. https://doi.org/10.1109/ICECS.2001.957449

[15] 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. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2114-2123. https://doi.org/10.1109/CVPR.2019.00222

[16] Wang, C.-Y., Yeh, I.-H., & Liao, H.-Y. M. (2024). YOLOv9: Learning what you want to learn using programmable gradient information. arXiv Preprint. https://doi.org/10.1007/978-3-031-72751-1_1

[17] Manisha-Sirsat. (2019). Retrieved from https://manisha-sirsat.blogspot.com/2019/04/confusion-matrix.html

[18] Padilla, R., Netto, S. L., & da Silva, E. A. B. (2020). A survey on performance metrics for object-detection algorithms. *2020 International Conference on Systems, Signals and Image Processing (IWSSIP)*, 237-242. https://doi.org/10.1109/IWSSIP48289.2020

[19] Sorgun, Ö. (2022). *Detection of prohibited objects using faster R-CNN in X-ray images*. Sakarya University of Applied Sciences, Graduate Education Institute.

Contact information:

Raşit KÖKER, Professor
(Corresponding author)
Sakarya University of Applied Sciences,
Faculty of Technology, Electrical and Electronics Engineering Department,
KemalpaşaMahallesi, 54050 Serdivan/SAKARYA
E-mail: rkoker@subu.edu.tr

Özcan SORGUN
Sakarya University of Applied Sciences,
Kemalpaşa Mahallesi, 54050 Serdivan/SAKARYA
E-mail: ozcansorgun17@gmail.com

Mustafa Çağri KUTLU, Assistant Professor
Sakarya University of Applied Sciences,
Faculty of Technology, Department of MechatronicsEngineering,
Kemalpaşa Mahallesi, 54050 Serdivan/SAKARYA
E-mail: mkutlu@subu.edu.tr

Mehmet DEMİR, Lecturer
Sakarya University of Applied Sciences Arifiye Vocational School,
Department of Electrical and Energy/Hybrid and Electric Vehicles Programme,
Fatih Mahallesi, 54580 Arifiye/SAKARYA
E-mail: mehmetdemir@subu.edu.tr