Zbigniew OMIOTEK

ORCID: 0000-0002-6614-7799

DOI: 10.15199/48.2025.05.36

OPEN ACCESS

Dangerous items' detection in surveillance camera images using Faster R-CNN

Wykrywanie niebezpiecznych przedmiotów na obrazach z kamer surveillance przy użyciu Faster R-CNN

***Abstract**. The Faster R-CNN with different backbone networks was used to detect dangerous objects in the study. The best results were obtained for the ResNet152 backbone. The mAP value was 85%, while the AP level ranged from 80% to 91%, depending on the item detected. An average real-time detection speed was between 11 and 13 FPS. Both the accuracy and speed of the model allow it to be recommended for use in public security monitoring systems aimed at detecting potentially dangerous objects.*

***Streszczenie.** W badaniach, do wykrywania niebezpiecznych obiektów wykorzystano sieć Faster R-CNN z różnymi sieciami szkieletowymi. Najlepsze wyniki uzyskano dla sieci szkieletowej ResNet152. Wartość mAP wyniosła 85%, natomiast poziom AP wahał się od 80% do 91%, w zależności od wykrywanego obiektu. Średnia prędkość wykrywania w czasie rzeczywistym wynosiła od 11 do 13 FPS. Zarówno dokładność, jak i szybkość modelu pozwalają rekomendować go do wykorzystania w systemach monitorowania bezpieczeństwa publicznego, mających na celu wykrywanie potencjalnie niebezpiecznych obiektów.*

Keywords: object detection, Faster R-CNN, deep learning, public safety.
Słowa kluczowe: wykrywanie obiektów, Faster R-CNN, uczenie głębokie, bezpieczeństwo publiczne.

Introduction

In the recent years, the problem of a growing number of incidents of violation of public security has become apparent. This is very well expressed by a statistic posted on the Statista website, showing the number of incidents involving active shooters in the United States between 2000 and 2022 (Fig. 1a). It should be noted that such incidents have also occurred with other types of items (e.g., knives, machetes, baseball bats). Considering the type of weapon used in the aforementioned acts of violence, various types of firearms (guns, rifles, shotguns) dominate here, but knives and various types of cutting tools (e.g., machetes) are also present. Figure 1b shows the number robberies in the United States in 2022, by weapon used.

In a report titled *A Study of Active Shooter Incidents in the United States Between 2000 and 2013*, the FBI identified 7 locations where the public is most vulnerable to security incidents [1]. These locations include: open space, commercial areas, educational environments, residences, government properties, houses of worship, health care facilities. Figure 2 shows the number of active shooter incidents that occurred at the mentioned locations in the United States in 2022. The summary in Figure 2 does not include incidents involving other types of weapons, which also occurred. A potential opportunity to prevent acts of violence is provided by intelligent vision systems using surveillance cameras installed in public places. The images acquired by these cameras can be analysed for dangerous objects brought in by people. If such items are detected, the appropriate services (police, facility security, etc.) can be notified early enough. In this way, the chance can be increased to prevent a potential act of violence. Great opportunities are offered here by deep convolutional neural networks, which are successfully used to detect various types of objects.

Among other applications, deep networks are used to automatically identify dangerous objects from X-ray images during airport security checks. Gao et al. proposed their own convolutional neural network model based on oversampling for detecting from an unbalanced dataset such objects as explosives, ammunition, guns and other weapons, sharp tools, pyrotechnics and others [3]. In contrast, Andriyanov used a modified version of the YOLOv5 network to detect ammunition, grenades and firearms in passengers' main baggage and carry-on luggage [4]. In the proposed solution, the initial detection is carried out by the YOLO model, and the results obtained are subjected to final classification by the VGG-19 network. A number of models have also been developed for use in monitoring systems for public places. Gawade et al. used a convolutional neural network to build a system designed to detect knives, small arms and long weapons. The accuracy of the built model was 85% [5]. Some authors propose their own original algorithms. One example is the PELSF-DCNN algorithm for detecting guns, knives and grenades [9]. According to the authors, the accuracy of their method is 97.5% and exceeds that of other algorithms. In contrast, Jang et al. proposed autonomous detection of dangerous situations in CCTV scenes using a deep learning model with relational inference [11]. The authors defined a baseball bat, a knife, a gun and a rifle as dangerous objects. The proposed method detects the objects and, based on relational inference, determines the degree of danger of the situations recorded by the cameras. Research on detecting dangerous objects is also being undertaken to protect critical infrastructure. Azarov et al. analysed existing real-time machine learning algorithms and selected the optimal algorithm for protecting people and critical infrastructure in large-scale hybrid warfare [12].

One of the challenges in object detection is the detector's ability to distinguish small objects during hand manipulation. Pérez-Hernández et al. proposed improving the accuracy and reliability of detecting such objects using a two-level technique [13]. The first level selects proposals for regions of interest, and the second level uses One-Versus-All or One-Versus-One binary classification. The authors created their own dataset (gun, knife, smartphone, banknote, purse, payment card) and focused on detecting weapons that can be confused by the detector with other objects. Another problem is the detection of

metal weapons in video footage. This is difficult because the reflection of light from the surface of such objects results in a blurring of their shapes in the image. As a result, it may be impossible to detect the objects. Castillo et al. developed an automatic model for detecting cold steel weapons in video images using convolutional neural networks [14]. The robustness of this model to illumination conditions was enhanced using so-called brightness-controlled preprocessing (DaCoLT) involving dimming and changing the contrast of the images during the learning and testing stages. The authors of an interesting paper in the field of dangerous object detection are Yadav et al. [15]. They made a systematic review of datasets and traditional and deep learning methods that are used for weapon detection.

- performance evaluation of the 5 different neural networks used as a backbone in the Faster R-CNN detector;
- determine the values of the most important hyperparameters of the Faster R-CNN training process;
- build a Faster R-CNN detector recommended for use in a public place monitoring system and test its effectiveness under real conditions;
- make available on the GitHub platform an implementation of the Faster R-CNN, which together with the shared dataset makes it possible to reproduce the experiments conducted.

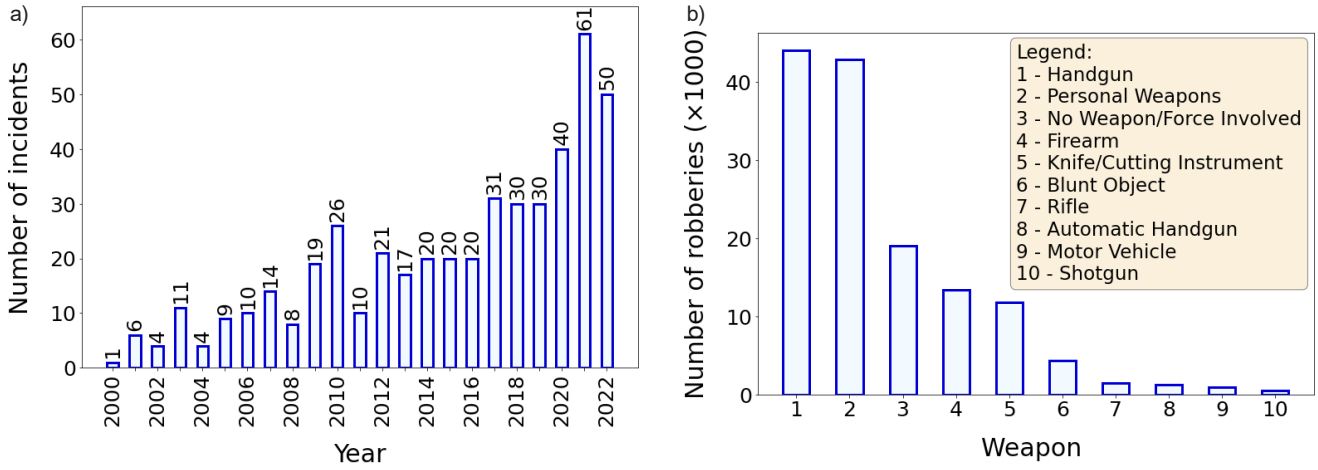


Fig. 1. Number of active shooter incidents in the United States from 2000 to 2022 (a) and number of robberies in the United States in 2022, by weapon used (b). Source: https://www.statista.com

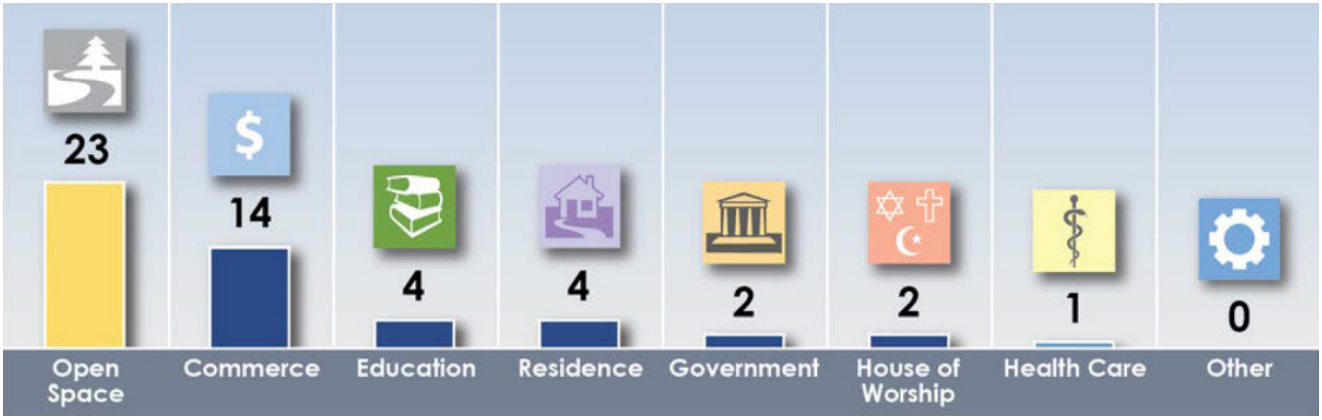


Fig. 2. Number of robberies in the United States in 2022, by location category [2]

The authors highlighted the problem of identifying the specific type of firearm used in an attack (so-called intra-class detection). The paper also compares the strengths and weaknesses of existing algorithms using classical and deep machine learning methods used to detect different types of weapons.

Most public-use facilities (schools, train stations, stores, etc.) already use functioning monitoring systems. Therefore, an interesting proposal may be a simple and low-cost solution to integrate into these systems a model based on the Faster R-CNN architecture, which would be trained to detect dangerous items in humans.

The most important contribution of this work is as follows:

- build and make available on the Zenodo platform a dataset dedicated to detecting dangerous items;

The Related Works section reviews works that has attempted to use Faster R-CNNs to detect dangerous objects. The Materials and Methods section briefly describes the Faster R-CNN used and characterises the object detection quality measures used and how the dataset was prepared. The Results section contains results of the training and evaluation processes. The Discussion section evaluates the results obtained and compares them with the results of other authors. It also describes the effects of testing the best model using a webcam. The entire work ends with a summary in the Conclusions section.

Related Works

There are many works that have addressed the problem of detecting dangerous items using the Faster R-CNN archi-

tecture. The following section will review selected solutions. An interesting proposal designed for automatic weapon detection is an ensemble of convolutional neural networks with different architectures. Haribharathi et al. used several pipelined convolutional neural networks [6]. According to the authors, this solution gives a 5% increase in accuracy, sensitivity and specificity compared to other solutions (RetinaNet, YOLOv7, SSD, etc.). Kambhatla et al. conducted a study on automatic detection of pistol, knife, revolver and rifle using YOLOv5 and Faster R-CNN [7]. Pruning and ensembling techniques were applied to YOLOv5 to increase their speed and efficiency. YOLOv5 models achieved the highest score of 78% with an inference speed of 8.1 ms. However, the Faster R-CNN models achieve the highest AP score of 89%. The work of Jain et al. implemented automatic weapon detection using SSD and Faster R-CNN algorithms [8]. The proposed implementation uses two types of datasets – images pre--labeled in automatic manner and manually labeled images. Both algorithms achieve good accuracy, but, the authors say, their application in real-world situations may be based on a trade-off between speed and accuracy. A novel system for automatic detection of handguns in video footage for surveillance and control purposes is also presented in [10]. The best results there were obtained for a model based on Faster R-CNN trained on its own database.

The authors report that in 27 out of 30 scenes, the model successfully activates the alarm after five consecutive positive signals with an interval of less than 0.2 seconds. The purpose of the system proposed by Vijayakumar et al. is to detect the presence of weapons, identify their type, and capture images of attackers [24]. A dataset consisting of 5 types of weapons was built: axe, knife, pistol, rifle and sword. Based on this dataset, the Faster R-CNN and YOLOv4 models were built. The YOLOv4 model provided a mAP score of 96.04% and 19 FPS with an average accuracy of 73%. The Faster R-CNN model achieved an average accuracy of 71%. An interesting solution is a weakly supervised object detection algorithm that learns to detect the orientation of bounding boxes based on frames aligned to the image axis [25]. This algorithm differs from existing solutions because, unlike them, it does not require the presence of oriented bounding boxes. The algorithm uses 2 stages of training. The goal of stage 1 is to predict frames aligned with the image axis, while stage 2 predicts the orientation of these frames. Between these two stages, the orientation proposal generation and ROI pooling modules extract features invariant to orientation. To evaluate the algorithm, the authors built a dataset of 11,000 pistol and rifle images. Hnoohom et al. used SSD MobileNet-V1, EfficientDet-D0 and Faster R-CNN Inception Resnet-V2 to detect weapons in a CCTV Surveillance System [26]. They conducted the experiments using 2 publicly available data sets: the IMFDB (Internet Movie Firearms Database) and *ARMAS dataset*. The authors obtained the best results for the Faster R-CNN model. The subject of weapon detection using CCTV is the work of González et al., in which the authors used images acquired with a CCTV camera and artificially generated images [27]. They used the Faster R-CNN model with ResNet50 as the backbone network, which allowed inference time of 90 ms. Based on this, the authors make conclusions about the impact of artificial datasets on the training of weapon detection models and point out the main limitations of such systems. Bhatti et al. conducted a study aimed at minimizing the number of false-positive and false-negative cases when detecting pistoles [28]. For this purpose, they built own dataset based on various repositories available on the Internet and self-recorded images. The pistol class included a pistol and a revolver, while the second one (confusion class) included a mobile phone, metal detector, selfie stick, wallet, purse, et al. On this basis, the authors tested many algorithms, including: VGG16, InceptionV3, Inception-ResnetV2, SSD--MobileNetV1, Faster R-CNN Inception-ResnetV2, YOLOv3 and YOLOv4. They obtained the best results for the YOLOv4 model, which yielded an F1 of 91% and an average precision of 91.73%. The problem of detecting guns and knives was also addressed by Fernandez-Carrobles et al. [29]. They proposed a weapon and knife detection system based on the Faster R-CNN model. The authors compared 2 approaches, in which they used GoogleNet and SqueezeNet architectures as the backbone network, respectively. For gun detection, the best result was obtained for the SqueezeNet architecture, achieving AP=85.44%. For knife detection, the GoogleNet architecture achieved an AP of 46.68%. Sagar et al. proposed a solution for detecting weapons from images captured by CCTV systems [30]. The authors created their own dataset consisting of pistols and rifles. On this basis, they compared the performance of a convolutional neural network, SSD, YOLOv3 and Faster R-CNN. Experimental results showed better performance of Faster RCNN compared to other architectures. The overall accuracy in this case was 99.1%.

Most of the previously presented results relate to datasets containing different categories of objects that are not strictly focused on the problem of detecting dangerous objects. In addition, the results presented are overestimated because a significant portion of the images in the collections used to build and test the models represent the objects themselves to be detected. They are not in the hand of any person, are large enough and perfectly visible. Consequently, the detector has little problem detecting such objects correctly and the reported detection precision is high. Such an approach misses the purpose of the research, which is to detect objects in the hands of people (potential attackers). Such objects are usually smaller, partially obscured and less visible, making them more challenging for the detector.

Against this background, this research stands out because it provides an insight into the construction of an object detector using a more specialised data set, which increases the trustworthiness and significance of the results obtained. The dataset was prepared in such a way that it contains items that are most often applied in acts of violence and violation of public safety, these are: baseball bat, gun, knife, machete and rifle. So, one can say that a dataset dedicated to the task of detecting potentially dangerous objects was built and used. It is also important that objects are presented with different quality, namely: clearly visible, partially obscured, small (longer viewing distance), blurred (worse sharpness) and poorly lit. As a result, the dataset better reflects real-world conditions and the results obtained are more trustworthiness. The built dataset is publicly available to a wide range of researchers who are potentially interested in using it in own research. An additional contribution of this work is to show

the possibility of using the popular Faster R-CNN architecture to detect dangerous objects by public place monitoring systems. The conducted experiments have shown that this kind of detector is characterized by low hardware requirements and high accuracy, which allows it to be recommended for the same or similar applications as those included in the subject of this research.

Materials and Methods

The purpose of the research the results of which are presented in this paper was to build a model suitable for use in monitoring systems for public places. The objects monitored by such systems are usually people who move relatively slowly within the observed area. Therefore, the frequently used operating speed of such systems, amounting to several FPS, is quite sufficient. Taking this into account, the basic requirement for the model was high accuracy in detecting objects, especially small objects. The assumption made in the study was that the model would detect a specific set of potentially dangerous objects, such as baseball bat, gun, knife, machete, rifle. Comparing the size of these objects with the size of the monitored scene, there is no doubt that they are small objects, occupying a negligible portion of the image frame. Taking into account the requirements formulated earlier, the Faster R-CNN was chosen. This was due to the fact that, on the one hand, this type of network copes well with the detection of small objects, and on the other hand, it allows to achieve a processing speed of several dozen FPS, which is sufficient for use in monitoring systems.

Faster R-CNN

The genesis of the Faster R-CNN is related to the development of the R-CNN by R. Girshick et al. [16]. This network offered higher accuracy compared to previous traditional object detection methods, but its main drawback was its low speed. For each image, the R-CNN generated approximately 2000 region proposals of various sizes, which were then subjected to feature extraction using a convolutional network [17–19]. In 2015, R. Girshick proposed to improve the R-CNN and introduced a new network called Fast R-CNN [20]. Instead of extracting features independently for each ROI, this type of network aggregated them in a single analysis of the entire image. In other words, the ROIs belonging to the same image shared computation and working memory. The Fast R-CNN detector proved to be more accurate and faster than previous solutions, but it still relied on the traditional method of generating 2000 region proposals (Selective Search method), which was a time-consuming process [21].

The problem of the temporal efficiency of the Fast R-CNN was the reason for the development of a new detector called Faster R-CNN by S. Ren et al. [22]. It improved the detection speed of Fast R-CNN by replacing the previously used Selective Search method with a convolutional network known as a Region Proposal Network (RPN). Figure 3 shows the architecture of the Faster R-CNN detector, and its operation can be described as follows:

1. A convolutional neural network takes an input image and generates a map of image features on its output.
2. The feature map is passed to the input of the RPN attached to the final convolutional backbone layer. The RPN acts as an attention mechanism for Fast R-CNNs. It tells you which region to look out for in relation to the potential presence of the object.
3. The feature map is then passed to the ROI pooling layer to generate a new map with a fixed size.
4. At the end of the process, the fully connected layers perform a region classification and a regression of the bounding box position.

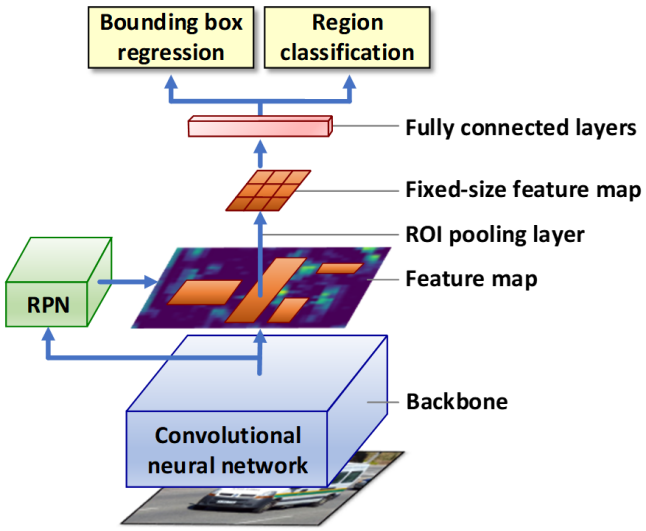


Fig. 3. Faster R-CNN [22]

The RPN takes input from a pre-trained convolutional network (e.g., VGG-16) and outputs multiple region proposals and predicted class labels for each of them. Region proposals are bounding boxes based on predefined shapes that are designed to speed up the region proposal operation. In addition, a binary prediction is used, indicating the presence of an object in a given region (the so-called objectness of the proposed region) or its absence. Both networks (RPN and backbone) are trained alternately, which allows you to tune parameters for both tasks at the same time.

The operation of the RPN network is explained in more detail in Fig. 4. The reception window is moved over the received feature map. For each position of this window, k ground-truth boxes of different sizes and proportions are created and anchored to the window, as shown in Fig. 4. In this way, multiple object suggestions are generated, each consisting of 4 coordinates of the ground-truth box and 2 values of the probability of the presence of an object in the box and its absence. When generating a large number of region proposals, many of them may overlap and correspond to the same object. To avoid this redundancy, the NMS (*Non Maximum Suppression*) method is used. This method orders the anchor boxes based on their objectivity index values and selects the top N of them with the highest scores. Thanks to this, the final selected proposals are accurate and do not overlap. The remaining anchor boxes are suppressed, and the selected boxes are considered possible region proposals. Then, these proposals are passed to the ROI pooling layer to create fixed-size feature maps. Further operation of the detector, based on fully interconnected layers is identical to that of the Fast R-CNN network [21].

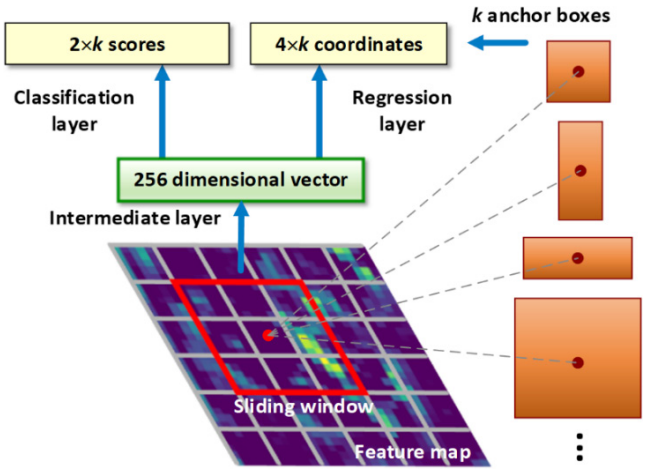


Fig. 4. Region Proposal Network (RPN) [22]

The introduction of the RPN has significantly reduced the number of regions proposed for further processing. This number was no longer equal to 2000, as in the case of Fast R-CNN, but was several hundred, depending on the complexity of the input images. This directly reduced training and prediction times and enabled the Faster R-CNN detector to be used in near real time.

Assessing the quality of object detectors

Most indexes for assessing the quality of object detection are based on two key concepts, these are prediction confidence and Intersection over Union (IoU). Prediction confidence is a probability estimated by a classifier that a predicted bounding box contains an object. IoU, on the other hand, is a measure based on the Jaccard index that evaluates the overlap between two bounding boxes – the predicted (B_p) and the ground-truth (B_{gt}). With IoU, it is possible to determine whether the detection is correct or not. The measure is the area of overlap between the predicted and ground-truth frames divided by the area corresponding to the sum of the two frames (Fig. 5).

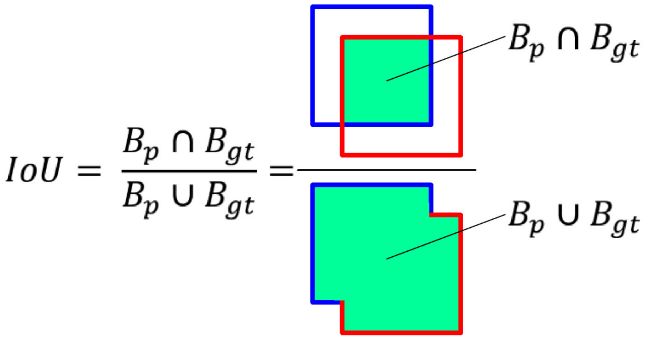


Fig. 5. Illustration of IoU parameter for predicted box B_p (blue) and ground-truth box B_{gt} (red)

Prediction confidence and IoU are used as criteria for determining whether object detection is true positive or false positive. Based on these, the following detection results can be determined:

True Positive (TP). A detection is considered true positive when it meets three conditions:

- the prediction confidence is greater than the accepted detection threshold (e.g., 0.5);
- the predicted class is the same as the class corresponding to the ground-truth box;
- the predicted bounding box has an IoU value greater than the detection threshold.

Depending on the metric, the detection threshold is usually set to 50%, 75% or 95%.

False Positive (FP). This is a false detection and occurs when either of the last two conditions is not met.

False Negative (FN). This situation occurs when the prediction confidence is lower than the detection threshold (no ground-truth box detected).

True Negative (TN). This result is not used. During object detection, there are many possible bounding boxes that should not be detected in the image. So, a TN value would mean all possible bounding boxes that were not correctly detected (there are a lot of them in the image).

The detection results allow calculating the values of 2 key quality measures, which are precision and sensitivity. **Precision** is an ability of the model to detect only correct objects. It is a ratio of the number of true positive (correct) predictions to the number of all predictions (1):

$$Precision = \frac{TP}{TP+FP} = \frac{TP}{all\ predictions} \tag{1}$$

Recall determines the model's ability to detect all valid objects (all ground-truth boxes). It is a ratio of the number of true positive (correct) predictions to the number of all ground-truth boxes (2):

$$Recall = \frac{TP}{TP+FN} = \frac{TP}{all\ ground-truth\ boxes} \tag{2}$$

Precision-recall curve. One way to evaluate the performance of an object detector is to change the level of detection confidence and plot a precision-recall curve for each class of objects. An object detector of a certain class is considered good if its precision remains high as the recall increases. This means that if the detection confidence threshold changes, the precision and recall will still remain high. Put a little differently, a good object detector detects only the right objects (no false-positive predictions means high precision) and detects all ground-truth boxes (no false-negative predictions means high recall).

Average precision. Precision-recall curves often have a zigzag course, running up and down and intersecting each other. Consequently, comparing curves belonging to different detectors on a single graph becomes a difficult task. Therefore, it is helpful to have an average precision (AP), which is calculated by averaging the precision values corresponding to sensitivities varying between 0 and 1. The average precision is calculated separately for each class. There are 2 approaches to calculating average precision. The first involves interpolating 11 equally spaced points. The second approach, introduced in the 2010 PASCAL VOC competition, involves interpolating all data points. The average precision is calculated for a specific IoU threshold, such as 0.5, 0.75, etc.

The 11-point interpolation approximates the precision-recall curve by averaging the precision for a set of 11 equally spaced recall levels [0, 0.1, 0.2,…, 1] (3):

$$AP = \frac{1}{11} \sum_{r \in \{0,\ 0{,}1,\ \ldots,\ 1\}} p_{interp}(r) \tag{3}$$

where the interpolated precision at a specific recall level r is defined as the maximum precision occurring for the recall level $r' \geq r$ (4):

$$p_{interp}(r) = p(r') \tag{4}$$

In the second method, instead of interpolation based on 11 equally spaced points, interpolation for all points is used (5):

$$AP = \sum_{i=1}^{n-1} (r_{i+1} - r_i) p_{interp}(r_{i+1}) \tag{5}$$

where $r_1, r_2, \ldots, r_n$ are the recall levels (in ascending order) at which the precision is interpolated (Fig. 6).

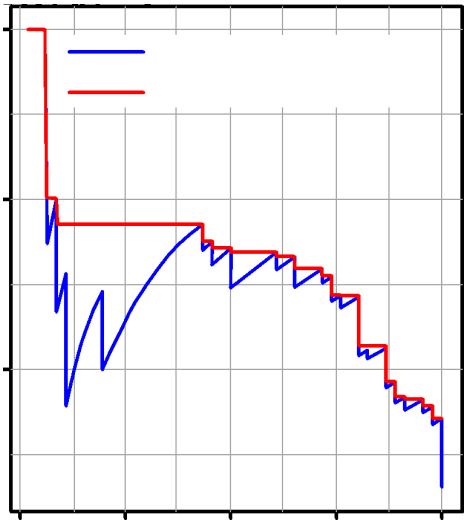


Fig. 6. Precision-Recall curves original and interpolated

Mean AP. Average precision covers only one class, while object detection mostly involves multiple classes. Therefore, mean average precision (mAP) is used for all K classes (6):

$$mAP = \frac{\sum_{i=1}^{K} AP_i}{K} \tag{6}$$

Inference time. Inference time is a time it takes for a trained model to make a prediction of new, previously unknown data on a single image. When detecting objects in video sequences, in addition to inference time, a second parameter is often used, namely the **frame rate per second** (FPS). This parameter indicates the frequency at which inference is performed for successive frames of the video stream. The choice of object detection model

depends on the specific requirements of the application. Some of them favour accuracy over speed (as in this research), while others prefer faster processing, especially in real-time applications. Consequently, researchers and developers often use FPS as a critical factor when deciding which model to implement for a particular use case. Be aware that the FPS value depends on the inference time, but also on the duration of additional operations, such as image capture, pre-processing, post-processing, displaying the results on the screen, etc.. The values of inference time and FPS depend on the architecture of the model and the hardware/software platform on which the model was run.

Preparing the dataset

In order to gather the dataset, a number of repositories available on the Internet that offer free access to the collected resources were searched. During this process, care was taken to ensure that the images presented a variety of situations recorded by surveillance cameras installed in public places. As a result, 5 categories of images were included in the dataset, as shown in Fig. 7. These are: clearly visible objects, partially covered objects, small objects, poor object sharpness and poor object illumination. As part of the image's preparation for the study, they were scaled (and cropped where necessary) so that the smaller side was no shorter than 600 pixels and the larger one was no longer than 1,000 pixels.

This operation was conditioned by the requirements of the Faster R-CNN. The next stage of data preparation was image labelling. This activity consisted of marking the detected objects on the images using bounding boxes. Label Studio was used for this purpose [23]. The image annotations were saved in the Pacal VOC format, according to which the description about each image is included in the corresponding XML file.

The full dataset contained 4,000 images. This set was randomly divided into training set (75% of the full set) and test set (25% of the full set). As a result, the training set contained 3,000 images, and the test set – 1,000. Figure 8 details the number of detected objects in each of the aforementioned sets. The dataset was made available on the Zenodo platform[1].

Results

Training process

During training, 5 architectures of the pre-trained backbone networks were used: ResNet50, ResNet101, ResNet152 and 2 VGG-16 networks differing in the pre-trained weights (VGG-16 Caffe and VGG-16 Pytorch). During the model's construction stage, 30 training epochs were used. While the first 15 epochs, the learning rate was 10^{-3}, and during the next 15 epochs its value was 10^{-4}. After each epoch, the mean average precision (mAP) of the model was calculated based on the test set. At the same time, the weights obtained in a given epoch were saved, so it was possible to select the best set of weights after the entire training process was completed. The implementation of the Faster R-CNN used in the experiments has been made available on the GitHub platform[2]. Table 1 collects the values of the more important hyperparameters used in the training process.

The images belonging to the training set were characterized by a great diversity. The detected objects were presented at different size scales, with different angles of rotation and different horizontal and vertical displacement within the image frame. In addition, the images were characterized by different levels of sharpness and brightness, and some of the detected objects were partially covered by other objects. Given these features and a relatively large number of objects belonging to each class, additional augmentation was abandoned at the model training stage.

A computer with Windows 11 64-bit operating system, Intel Core i7–12650H 2.70 GHz processor and 32 GB RAM was used to train the models. PyTorch 2.1.2 platform and the Python 3.10 programming language were employed. Calculations were performed using the GPU (NVIDIA GeForce RTX

[1] https://zenodo.org/records/11582054

[2] https://github.com/zomiotek/Faster-R-CNN-for-Dangerous-Items--Detection

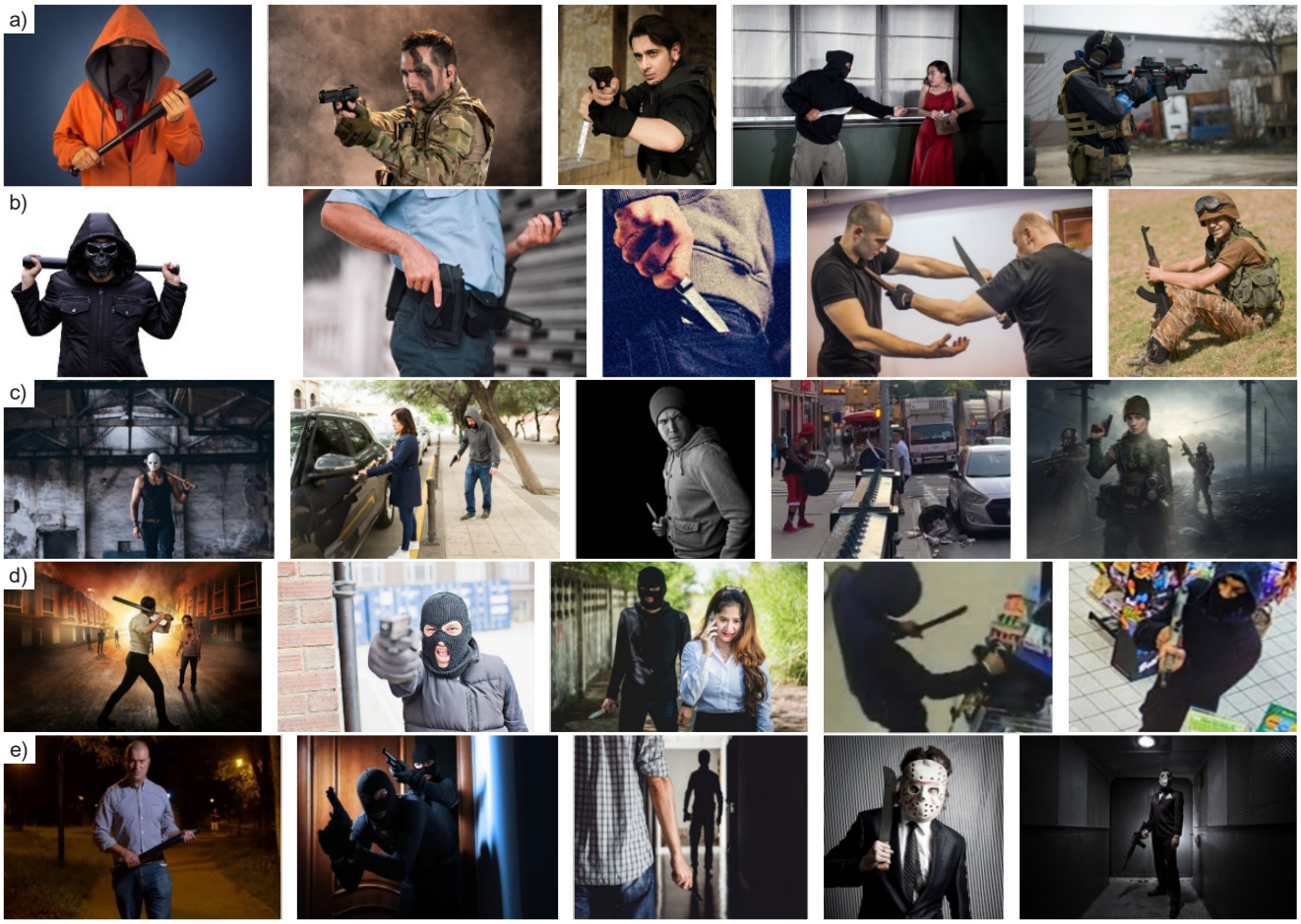


Fig. 7. Categories of images included in the dataset: a) objects clearly visible; b) objects partially covered; c) small objects; d) poor objects sharpness; e) poor objects illumination

Table1. Hyperparameter values used during the models training process

Name	Value
Epochs	30
RPN mini-batch	256
Proposal batch	128
Object IoU threshold	0.5
Background IoU threshold	0.3
Dropout	0.0
Loss function	multi-task loss function*
Optimizer	stochastic gradient descent (SGD)
Learning rate	10^{-3} (epochs 1–15), 10^{-4} (epochs 16–30)
Momentum	0.9
Weight-decay	5×10^{-4}
Epsilon	10^{-7}
Augmentation	–

* multi-task loss function combines classification and regression losses. The meaning of the selected hyperparameters: *RPN mini--batch* – the number of ground truth anchors sampled for training at each step; *proposal batch* – the number of region proposals to sample at each training step; *object IoU threshold* – object IoU threshold between an anchor and a ground truth box above which an anchor is labelled as an object (positive) anchor; *background IoU threshold* – background IoU threshold below which an anchor is labelled as background (negative).

3060 Laptop GPU, 6 GB GDDR6), the CUDA 11.8 platform and the cuDNN 8.9.6 library. Table shows the total training time of each of the five models built.

Model evaluation

Based on the test set, 5 models differing in backbone network were evaluated. The test set was balanced, as was the full dataset. The percentage of each object class in it was about 20% (Fig. 8c). Figure 9 shows precision-recall plots for each object category. The performance of a given model is better the longer a high precision value is maintained with an increasing recall value. In the graph, this is manifested in such a way that the precision-recall curve runs closer to the point with coordinates (1, 1). The graphs shown in Fig. 9 indicate that when detecting baseball bat, knife and machete, the best performance was achieved by models based on ResNet101 and Resnet152. In contrast, for gun and rifle, the model based on ResNet152 was the best.

Figure 10 shows bar graphs reflecting the average precision of detection of different object categories by each model. The gun was detected with the highest precision by ResNet152 ($AP_{0.5}$=91.3%). The same object was detected with slightly lower precision by ResNet101 ($AP_{0.5}$=89.7%). Third place went to gun detection accuracy by ResNet50 ($AP_{0.5}$=89.5%). The worst results were for a machete detected by VGG16_PyTorch ($AP_{0.5}$=68.5%) and a knife detected by VGG16_PyTorch ($AP_{0.5}$=68.3%).

Bar graphs showing the value of the mean average precision of detection of all 5 object classes are shown in Fig. 11.

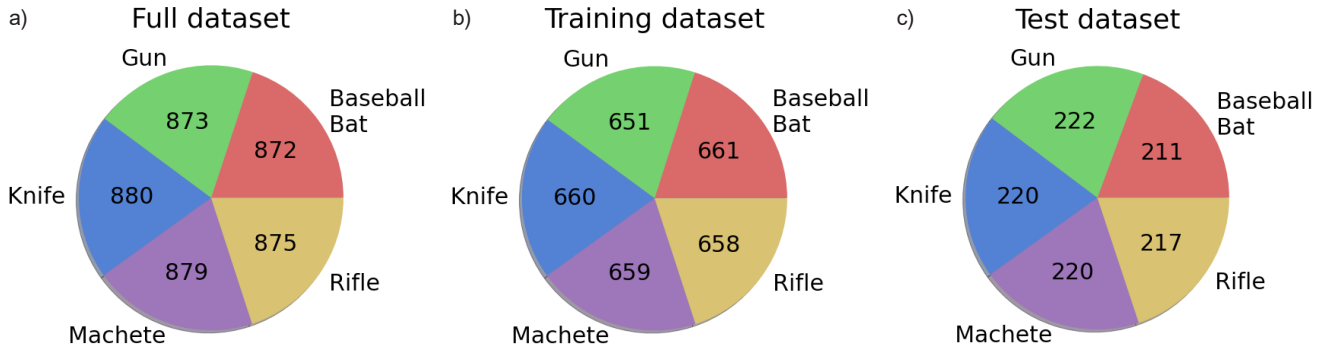


Fig. 8. The number of detected objects in the: a) full; b) training and c) test datasets. The share of a given category of objects in each set is 20%

Table 2. Total training time

Backbone	Training time
VGG-16 Caffe	3h 43m 10s
VGG-16 Pytorch	4h 10m 39s
ResNet50	3h 42m 9s
ResNet101	4h 17m 10s
ResNet152	6h 25m 13s

The best result was achieved by ResNet152 (mAP=85%), a slightly worse mAP value (84.5%) was achieved by ResNet101. The average precision of the other models was: 81.8% (ResNet50), 78.3% (VGG16_Caffe) and 77.4% (VGG16_PyTorch).

Model evaluation was performed on the same computer as the network training (computer specifications are given in the *Training process* section). The average inference time per image frame ranged from 74 ms for VGG-16_Caffe and VGG-16_Pytorch to 101 ms for ResNet152, depending on the topological depth of the backbone network used (Fig. 12).

Discussion

Assessment of the results obtained

Figure 13 shows an example of the results obtained using the ResNet152 model. The analysed images could be divided into several categories, depending on the presentation of the detected objects. The first category included images with objects that were clearly visible (Fig. 13a), which were detected with 100% accuracy, regardless of their size. Objects that were partially hidden in the hand (gun, knife) or partially obscured by body parts of the subject were detected with similar accuracy (Fig. 13b). In the images belonging to the third category, the detected objects were presented on a smaller scale because they were further away from the camera (Fig. 13c). Again, the model did very well in detecting them, as shown by the predicted bounding boxes in red. The next type of images had poorer sharpness, as a result of which the detected objects were less clear than in the other images (Fig. 13.d). Detection confidence here was not 100%, as for the previous categories of images. There were also greater differences in localization between the predicted bounding boxes and the ground-truth boxes. However, despite the more difficult conditions, the model performed very well in detecting all classes of objects. In most cases, detection confidence exceeded 90%. The last category of images presented the detected objects in poorer lighting, resulting in darker images and less visible objects (Fig. 13e). Despite the more difficult conditions, the model easily detected objects that were faintly visible, small (gun, knife) and at a greater distance from the camera. The results showed that the advantage of the built model was that it was highly effective in detecting objects, regardless of the conditions of their presentation, such as the sharpness and brightness of the image, the size of the objects or their partial obscuration by other objects.

In order to carry out more accurate diagnostics of the model, a confusion matrix was constructed (Fig. 14.). It allows to determine the type and number of errors made by the model. The most errors were made for knives and machetes. When detecting these items, 14 knives were recognized as machetes and 15 machetes were classified as knives. In this case, there were a number of images showing long knives and short machetes, which could pose a challenge for the detector. The confusion matrix also included cases where objects were detected when they were not actually present in the image (false positives) and cases where existing objects were not detected (false negatives). The number of false--positive cases was 112, while the number of false-negative cases was much lower at 37. The highest number of false detections was observed for knives (32), machetes (27) and rifles (22). In order to reduce the number of such false detections, the number of training cases will be increased in the course of further research, and augmentation will be used, which was abandoned in the present study. The confusion matrix also made it possible to calculate the recall of the model. The value of this parameter was as follows: baseball bat – 95.3%, gun – 94.6%, knife – 90.0%, machete – 85.5%, rifle – 88.5% (Fig. 14b).

Comparison of results with those of other authors

Table 3 summarises the dangerous object detection results obtained by different authors using the Faster R-CNN. Unfortunately, for various reasons, it is difficult to compare own results with them. Other authors have used different datasets in their studies. Some of them are available on the Internet, while others were prepared specifically for the research in progress. In turn, the images used in the experiments varied in terms of size, type (photographs,

video) and presentation of the objects detected (objects occupying the entire surface of the image and objects presented in real life – held in a person's hand). Different measures of performance were also used (accuracy, AP, mAP,

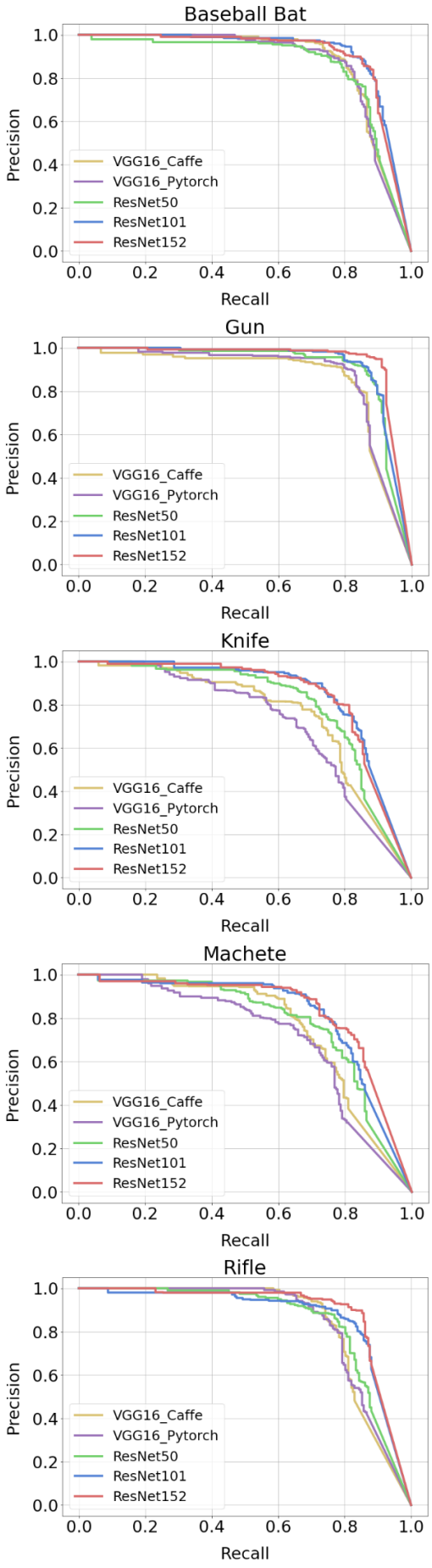


Fig. 9. Precision-recall plots for each category of detected objects: a) baseball bat; b) gun; c) knife; d) machete; e) rifle

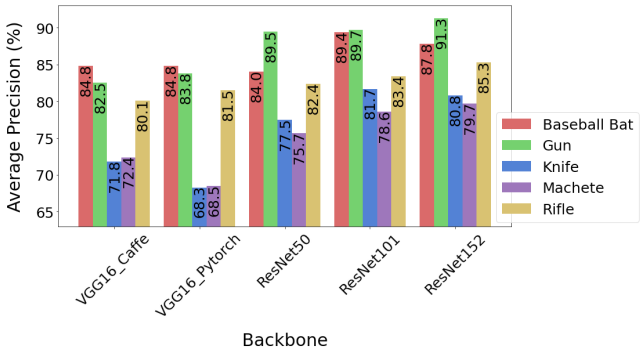


Fig. 10. Average precision of the object detection for individual backbone networks. The average precision was measured for an IoU threshold of 0.5

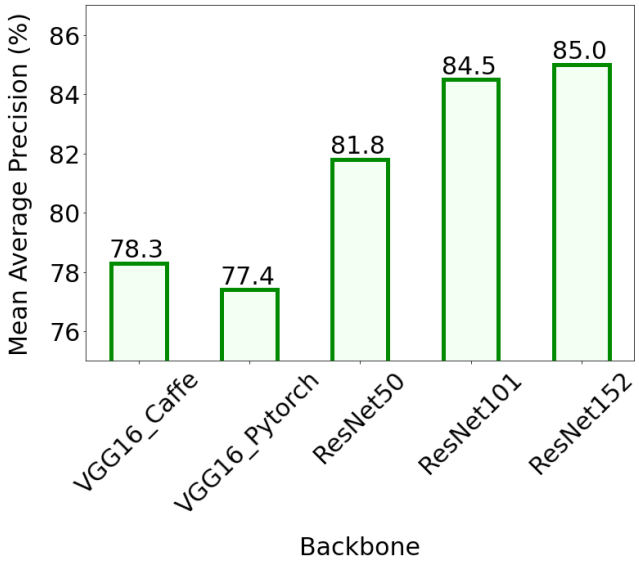


Fig. 11. Mean average precision for individual backbone networks

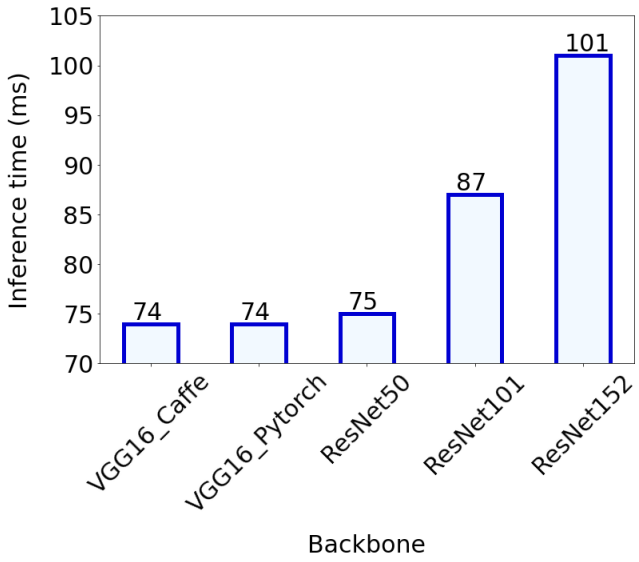


Fig. 12. Inference time averaged over 1000 test images

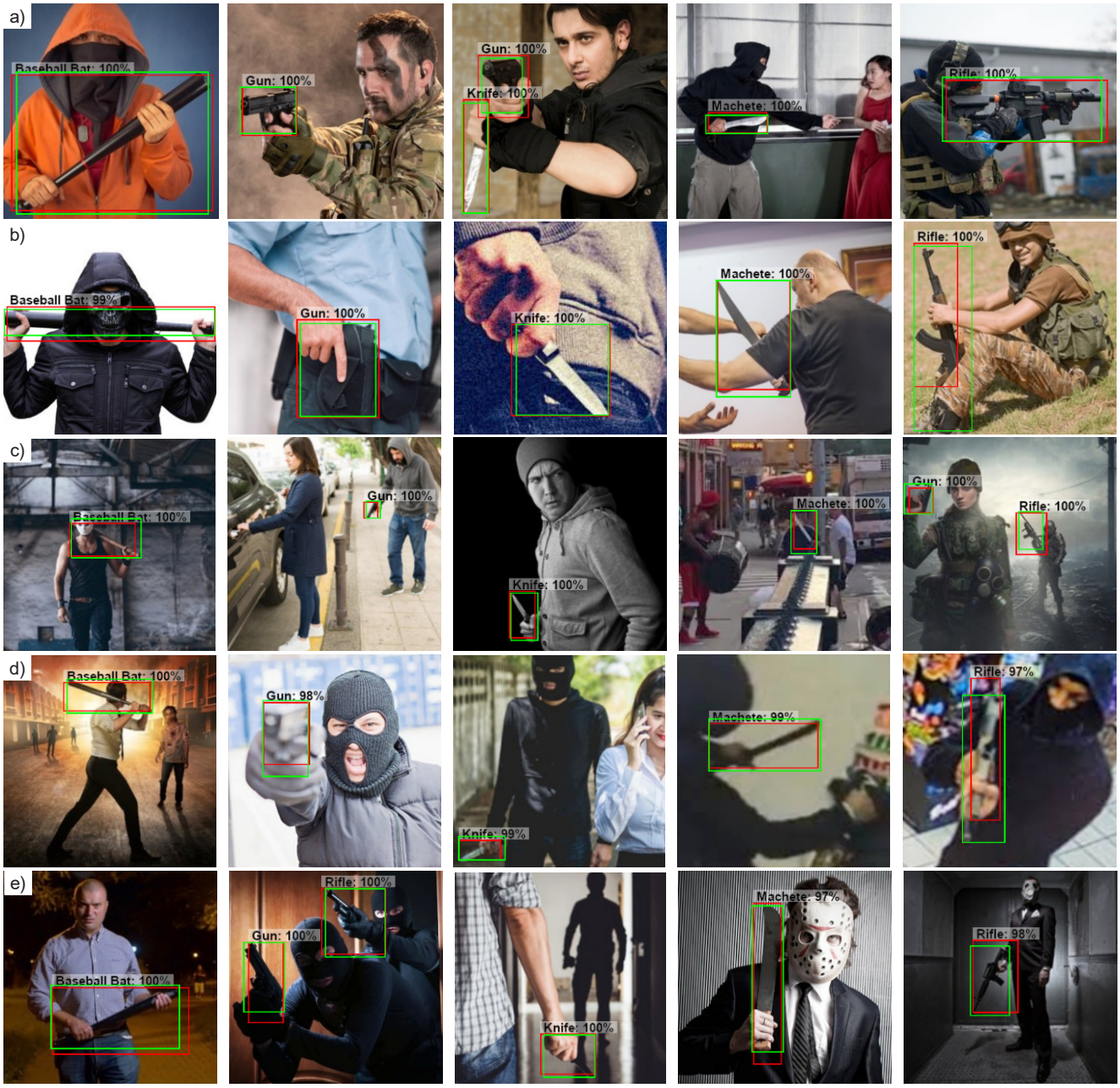
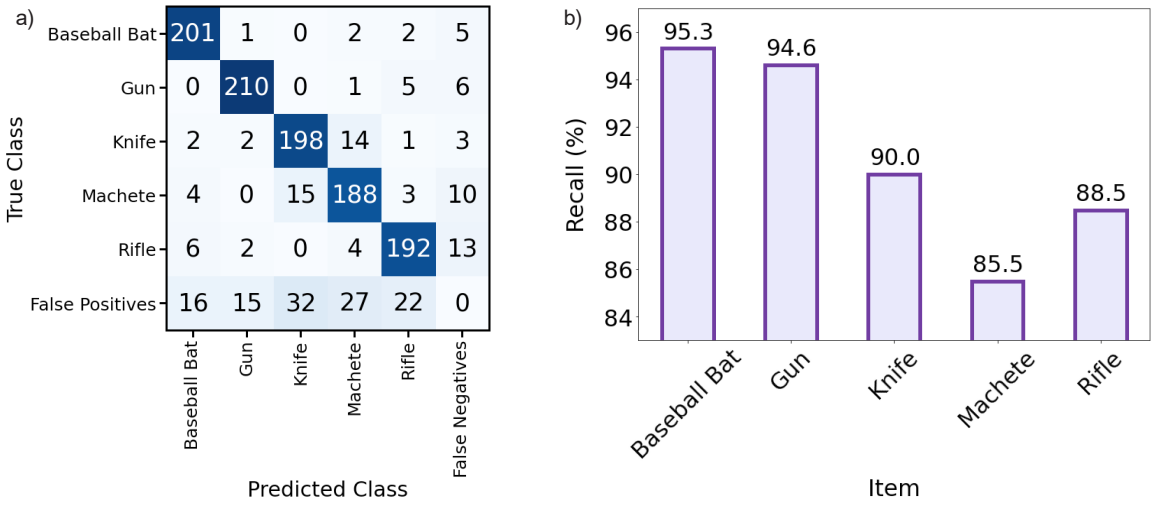


Fig. 13. Example results of detecting objects presented with different quality: a) objects clearly visible; b) objects partially covered; c) small objects; d) poor objects sharpness; e) poor objects illumination. Ground-truth bounding boxes are drawn in green and prediction results are marked in red. The images have been cropped so that it is easier to compare the location of ground-truth and predicted bounding boxes



F1-score and others). However, the most commonly used measures were precision, mAP, recall and mAR with a standard IoU threshold of 0.50. Therefore the aforementioned measures were used in the summary in Table 3.

The largest number of results available relates to gun detection, as this is the most commonly used weapon in assaults and acts of violence, according to statistics. The obtained precision value for this case (AP=91.3%) is in second place. The best precision for gun detection was obtained by Vijayakumar et al. – this was 96.6% [24]. Unfortunately, the dataset used in this study leaves much to be desired. One of the problems is that the authors do not provide information about the number of individual classes (detected objects), but only the number of images. According to the examples given by the authors, in most cases, one image presented one object to be detected. According to the data, the full set contained only 120 images of pistols, of which 96 images were used for training and 24 for testing. In addition, a significant proportion of the images (60%) depicted guns without a real background, and only 40% showed them in real conditions (held in the hand). Let us remind you that collection used contained 873 pistols (training set – 651, test set – 222). Both the number and structure of the set of images used give grounds for suspicion that the obtained result is overestimated. Regarding the recall of gun detection, the best result (AR=100%) was obtained by Olmos et al. [10] and González et al. [27]. Own result, with AR=94.6%, is in second place. It is difficult to compare these results, because Olmos et al. and González et al. obtained a recall of 100% for binary models that were designed to detect only pistols (or pistols and rifles). The model used in the research is not binary but multiclass (it detects 5 classes of objects), and the performance of such a model is usually lower than that of a binary one. In such a situation, the obtained recall of 94.6% is assessed as very high. Far fewer results are available for the detection rifles. A rifle detection precision of 85.3% obtained in own study is a second best one in the presented list. The best precision (AP=100%) was obtained by Vijayakumar et al. [24]. However, it seems that the value of the precision given by the authors is overestimated due to the very limited data set. The full collection contained only 135 images of rifles, of which 108 were used for training and 27 for testing. For comparison, a collection used contained 875 rifles (training set – 658, test set – 217). The detection recall of rifles was 88.5% in this research. Vijayakumar et al. [24] achieved a higher recall of 96% in this case. Unfortunately, this result is not very reliable due to the size of the dataset used (as in the case of precision). In the case of knife detection precision, by far the best result (AP=80.8%) was obtained. It is far superior to the results obtained by other authors, which were AP=55.8% (Vijayakumar et al. [24]) and AP=46.7% (Fernandez-Carrobles et al. [29]). Similarly, the best knife detection recall of 90.0% was achieved. The best result obtained by other authors in this case was 52% (Vijayakumar et al. [24]). However, it was not possible to find research results on the use of the Faster R-CNN for the detection of baseball bats and machetes. This demonstrates the unique nature of the dataset used in the study and the results obtained. In terms of average precision and average sensitivity for detecting a set of different objects, the results obtained were the highest (mAP=85%, mAR=90.6%).

Application of the results obtained to the prediction of new images

The model that proved most effective at the evaluation stage (ResNet152) was selected for further testing involving image prediction using the camera. The experiments used the same computer used for model training and evaluation, and a Tracer HD WEB008 webcam acting as a surveillance

Table 3. Results of similar studies using Faster R-CNN. A standard IoU threshold of 0.50 was used for all measures

No. in Refs.	Backbone	$mAP_{0.5}$ (%)	$AP_{0.5}$ (%)	$mAR_{0.5}$ (%)	$AR_{0.5}$ (%)
Own results	ResNet152	85.0 (baseball bat, gun, knife, machete, rifle)	87.8 (baseball bat) 91.3 (gun) 80.8 (knife) 79.7 (machete) 85.3 (rifle)	90.6 (baseball bat, gun, knife, machete, rifle)	95.3 (baseball bat) 94.6 (gun) 90.0 (knife) 85.5 (machete) 88.5 (rifle)
[6]	CNN	–	84.7 (gun)	–	86.9 (gun)
[8]	CNN	84.6 (gun, rifle)	–	–	–
[10]	VGG-16	–	84.2 (gun)	–	100 (gun)
[24]		80.5 (axe, gun, knife, rifle, sword)	96.6 (gun) 55.8 (knife) 100 (rifle)	65.8 (axe, gun, knife, rifle, sword)	61 (gun) 52 (knife) 96 (rifle)
[25]	VGG-16	79.8 (gun, rifle)	80.2 (gun) 79.4 (rifle)	–	–
[26]	Inception-ResNetV2	–	79.3 (gun)	–	68.6 (gun)
[27]	ResNet50	–	88.12 (gun)	–	100 (gun)
[28]	Inception-ResNetV2	–	86.4 (gun)	–	89.25 (gun)
[29]	SqueezeNet	–	85.4 (gun)	–	–
	GoogleNet	–	46.7 (knife)	–	–

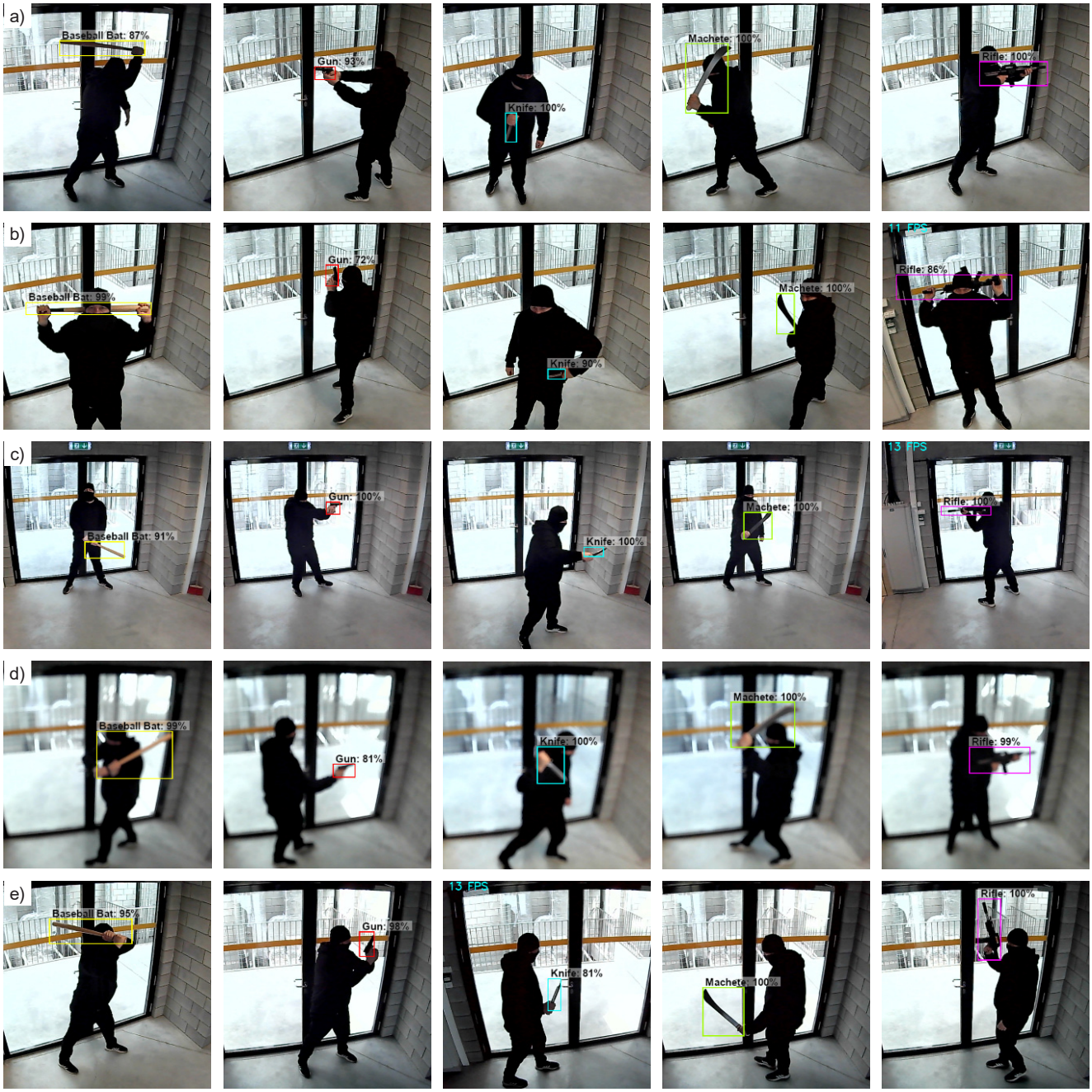


Fig. 15. The selected image frames recorded during detection of dangerous items in people entering the room: a) objects clearly visible; b) objects partially covered; c) small objects; d) poor objects sharpness; e) poor objects illumination. For cases a), b), d) and e) the camera distance and installation height were 2.5 m. For case c) the camera distance was increased to 5 m. The original image frame size was 800×600 pixels. In order to better present the detected objects, they were cropped to 600×600 pixels

camera. The scenario of the experiment involved monitoring people entering the room and detecting potentially dangerous objects in them. Both the distance of the camera from the entrance door and its installation height were 2.5 meters. Figure 15 shows selected frames of the recorded image.

During recording, the frames were scaled to 800×600 pixels so as to meet the Faster R-CNN model's minimum input image size requirements. At this resolution, the average processing speed was 11 to 13 FPS. Such a speed is sufficient for use in public space surveillance systems, where the typical processing speed of image frames is usually a dozen FPS. During the experiment, the vast majority of objects were correctly detected, and the image was played back with satisfactory smoothness. Despite the correct operation of the model, there were image frames where objects were not detected. This was mainly related to the greater distance from the camera and poor lighting. During further research, an attempt will be made to eliminate these problems by increasing the number of training images taking into account the situations described above and the augmentation technique. Selected excerpts from the course of the experiment are shown in a video made available on the Zenodo platform[3].

Conclusions

As a result of the study, 5 models were built, of which the most effective one, based on the ResNet152 backbone,

[3] https://zenodo.org/records/12736924

achieved a mAP value of 85%. This is a very good result, if we take into account the specifics of the dataset used. It consisted (in similar proportions) of images in which the detected objects were: clearly visible, partially obscured, small, indistinct (partially blurred) and faintly visible (dark). This structure of the collection was a factor in increasing the difficulty of the object detection task. The evaluation conducted on the test set and the prediction of new images using an ordinary webcam showed that the ResNet152 model performs very well in detecting objects, regardless of the quality of the input image. Also, the processing speed of video frames is satisfactory. In the computer system used in the experiments, the value of the mentioned parameter was between 11 and 13 FPS, which gives the possibility of practical application of the proposed solution. The obtained results entitle to recommend the Faster R-CNN with the ResNet152 backbone network for use in public monitoring systems, whose task is to detect specific objects (e.g., potentially dangerous objects) brought by people to the premises of various facilities. The research used a specific set of these objects (baseball bat, gun, knife, machete, rifle), but according to the goal set for the monitoring system, this set can be expanded or modified accordingly.

***Author**: dr hab. inż. Zbigniew Omiotek, Politechnika Lubelska, Katedra Elektroniki i Technik Informacyjnych, ul. Nadbystrzycka 38A, 20–618 Lublin, E-mail: z.omiotek@pollub.pl.*

REFERENCES

[1] Blair J.P., Schweit K.W., A Study of Active Shooter Incidents, 2000–2013. *Texas State University and Federal Bureau of Investigation, U.S. Department of Justice, Washington D.C.,* (2014)
[2] Active Shooter Incidents in the United States in 2022. *Federal Bureau of Investigation, U.S. Department of Justice, Washington, D.C., and the Advanced Law Enforcement Rapid Response Training (ALERRT) Center at Texas State University,* (2023).
[3] Gao Q., Li Z., Pan J., A Convolutional Neural Network for Airport Security Inspection of Dangerous Goods. *IOP Conf. Ser.: Earth Environ. Sci.,* 252 (2019).
[4] Andriyanov N., Deep Learning for Detecting Dangerous Objects in X-rays of Luggage. *Eng. Proc.,* 33 (2023), No. 20.
[5] Gawade S., Vidhya R., Radhika R., Automatic Weapon Detection for surveillance applications. *Proc. of the International Conference on Innovative Computing & Communication (ICICC) 2022*, (2022).
[6] Haribharathi S., Vijay Arvind R., Pawan Ragavendhar V., Balamurugan G., Novel Deep Learning Pipeline for Automatic Weapon Detection. *arXiv: 2309.16654v1 [cs.CV],* (2023).
[7] Kambhatla A., Khaled A.R., Real Time Deep Learning Weapon Detection Techniques For Mitigating Lone Wolf Attacks. *International Journal of Artificial Intelligence and Applications (IJAIA),* 14 (2023), No. 4.
[8] Jain H., Vikram A., Mohana Kashyap A., Jain A., Weapon Detection using Artificial Intelligence and Deep Learning for Security Applications. *Proc. of the International Conference on Electronics and Sustainable Communication Systems (ICESC 2020),* (2020).
[9] Dugyala R., Reddy M.V.V., Reddy Ch.T., Vijendar G., Weapon Detection in Surveillance Videos Using YOLOv8 and PELSF-DCNN. *4th International Conference on Design and Manufacturing Aspects for Sustainable Energy (ICMED-ICMPC 2023),* 391 (2023).
[10] Olmos R., Tabik S., Herrera F., Automatic handgun detection alarm in videos using deep learning. *Neurocomputing*, 275 (2018), 66–72.
[11] Jang S., Battulga L., Nasridinov A., Detection of Dangerous Situations using Deep Learning Model with Relational Inference. Journal of Multimedia Information System, 7 (2020), No. 3, 205–214.
[12] Azarov Iv., Gnatyuk S., Aleksander M., Azarov Il., Mukasheva A., Real-time ML Algorithms for The Detection of Dangerous Objects in Critical Infrastructures. *4th International Workshop on Intelligent Information Technologies and Systems of Information Security* 2023, 3373 (2023), 217–226.
[13] Pérez-Hernández F., Tabik S., Lamas A., Olmos R., Fujita H., Herrera F., Object Detection Binary Classifiers methodology based on deep learning to identify small objects handled similarly: Application in video surveillance. *Knowledge-Based Systems*, 194 (2020).
[14] Castillo A., Tabik S., Pérez F., Olmos R., Herrera F., Brightness guided preprocessing for automatic cold steel weapon detection in surveillance videos with deep learning. *Neurocomputing*, 330 (2019), 151–161.
[15] Yadav P., Gupta N., Sharma P.K., A Comprehensive Study towards High-level Approaches for Weapon Detection using Classical Machine Learning and Deep Learning Methods. *Expert Systems with Applications*, 212 (2023).
[16] Girshick R., Donahue J., Darrell T., Malik J., Rich feature hierarchies for accurate object detection and semantic segmentation. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, (2014), 580–587.
[17] Zhao Z.-Q., Zheng P., Xu S.-t., Wu X., Object detection with deep learning: a review. *IEEE Trans. Neural Netw. Learn. Syst.*, 30 (2019), No. 11, 3212–3232.
[18] Pulkit S., Introduction to object detection algorithms. https://www.analyticsvidhya.com/blog/2018/10/a-step-by-step-introduction-to-the--basic-object-detection-algorithms-part-1/, (2018). Accessed on 10.06.2024.
[19] Uijlings J.R.R., van de Sande K.E.A, Gevers T., Smeulders A.W.M., Selective Search for Object Recognition. *International Journal of Computer Vision,* 104 (2013), No. 2, 154–171.
[20] Girshick R., Fast r-cnn. *Proceedings of the IEEE International Conference on Computer Vision,* (2015), 1440–1448.
[21] Arulprakash E., Aruldoss M., A study on generic object detection with emphasis on future research directions. *J. King Saud Univ., Comput. Inf. Sci.,* (2021).
[22] Ren S., He K., Girshick R., Sun J., Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *arXiv: 1506.01497 [cs.CV],* (2016).
[23] Label Studio project homepage. https://labelstud.io/. Accessed on 10.06.2024 r..
[24] Vijayakumar K.P., Pradeep K., Balasundaram A., Dhande A., R-CNN and YOLOV4 based Deep Learning Model for intelligent detection of weaponries in real time video. *Mathematical Biosciences and Engineering*, 20 (2023), No. 12, 21611–21625.
[25] Iqbal J., Munir M.A., Mahmood A., Ali A.R., Ali M., Leveraging Orientation for Weakly Supervised Object Detection with Application to Firearm Localization. *arXiv: 1904.10032v2 [cs.CV],* (2021).
[26] Hnoohom N., Chotivatunyu P., Maitrichit N., Sornlertlamvanich V., Mekruksavanich S., Jitpattanakul A., Weapon Detection Using Faster R-CNN Inception-V2 for a CCTV Surveillance System. *2021 25th International Computer Science and Engineering Conference (ICSEC),* (2021), 400–405.
[27] González J.L.S., Zaccaro C., Alvarez-Garcia J.A., Morillo L.M.S., Caparrini F.S., Real-time gun detection in CCTV: An open problem. *Neural Networks,* 132 (2020), 297–308.
[28] Bhatti M.T., Khan M.G., Aslam M., Fiaz M.J., Weapon Detection in Real-Time CCTV Videos using Deep Learning. *IEEE Access,* 9 (2021).
[29] Fernandez-Carrobles M.M., Deniz O., Maroto F., Gun and Knife Detection Based on Faster R-CNN for Video Surveillance. *Pattern Recognition and Image Analysis. IbPRIA 2019. Lecture Notes in Computer Science*, (2019).
[30] Sagar D.M.C., Rajesh Y., Weapon Detection Using Deep learning model and Artificial Intelligence. *ZKG International*, 8 (2023), No. 1, 1419–1428.