Neural Networks 132 (2020) 297–308

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

Real-time gun detection in CCTV: An open problem

Jose L. Salazar González [a], Carlos Zaccaro [a], Juan A. Álvarez-García [a,*], Luis M. Soria Morillo [a], Fernando Sancho Caparrini [b]

[a] Dpto. de Lenguajes y Sistemas Informáticos, Universidad de Sevilla, 41012, Sevilla, Spain
[b] Dpto. de Ciencias de la Computación, Universidad de Sevilla, 41012, Sevilla, Spain

ARTICLE INFO

Article history:
Received 16 April 2020
Received in revised form 14 August 2020
Accepted 14 September 2020
Available online 17 September 2020

Keywords:
Deep learning
Convolutional neural network
Weapon detection
Feature Pyramid Network
Synthetic data
Data augmentation

ABSTRACT

Object detectors have improved in recent years, obtaining better results and faster inference time. However, small object detection is still a problem that has not yet a definitive solution. The autonomous weapons detection on Closed-circuit television (CCTV) has been studied recently, being extremely useful in the field of security, counter-terrorism, and risk mitigation. This article presents a new dataset obtained from a real CCTV installed in a university and the generation of synthetic images, to which Faster R-CNN was applied using Feature Pyramid Network with ResNet-50 resulting in a weapon detection model able to be used in quasi real-time CCTV (90 ms of inference time with an NVIDIA GeForce GTX-1080Ti card) improving the state of the art on weapon detection in a two stages training. In this work, an exhaustive experimental study of the detector with these datasets was performed, showing the impact of synthetic datasets on the training of weapons detection systems, as well as the main limitations that these systems present nowadays. The generated synthetic dataset and the real CCTV dataset are available to the whole research community.

1. Introduction

Nowadays, many scenarios require object detection to perform diverse tasks. Some only need to detect objects that represent a large part of the scene, yet others need to detect multiple objects of different sizes. When this happens, results are different depending on the size of the object, being small objects the ones that obtain the worst results. These results are reflected in challenges such as COCO, ImageNet, or Open Images, where small objects obtain approximately 38% less precision than medium objects. This problem is mainly because small objects contain fewer pixels and have fewer occurrences, either for not being tagged or not being represented.

The lack of precision when detecting small objects is a problem for sectors such as security, where it is necessary to detect an object as small as a handgun among a group of people, which may represent a scale of less than 3% of the scene. This study will analyze this case as part of the VICTORY[1] (Enríquez, et al., 2019; Salazar, Soria, Álvarez-García, Enríquez, & Jiménez, 2019) project, which aims to alert security personnel as soon as possible of a potential threat is detected, so the detector requires the proper speed to be able to adapt the weapon detection alert to a real-time CCTV.

This problem has been previously approached by modifying the network architecture to apply object detection in aerial and satellite images (Etten, 2018; Sommer, Schuchert, & Beyerer, 2017), increasing image resolution by adding Perceptual Generative Adversarial Networks (Li, et al., 2017), or manually adding small objects to the scenes (Kisantal, Wojna, Murawski, Naruniec, & Cho, 2019). Although these solutions improve the accuracy of small objects, they are focused on specific cases such as traffic signs or satellite images, which limits the area of applicability.

Another problem when applying an object detection algorithm to a particular problem is the lack of data available, given the amount of manual work required to tag each object in images. However, we can find larger general datasets that are often used to pre-train models. Among the most used open datasets are: MS-COCO (Lin, et al., 2014), a large-scale object detection, segmentation, and captioning dataset containing more than 200,000 labeled images and 80 object categories; ImageNet (Deng, et al., 2009), an image database organized according to the WordNet hierarchy (Miller, 1998) with 14,197,122 images and 21,841 synsets currently indexed; and Open Images (Kuznetsova, et al., 2020) with around 9 million annotated images including bounding boxes, segmentation masks, and visual relationships.

A common practice and with generally positive results is to apply data augmentation to datasets with different transformations to increase the number of data by a certain percentage.

* Corresponding author.
E-mail addresses: jsalazar@us.es (J.L. Salazar González), czaccaro@us.es (C. Zaccaro), jaalvarez@us.es (J.A. Álvarez-García), lsoria@us.es (L.M. Soria Morillo), fsancho@us.es (F. Sancho Caparrini).

[1] VICTORY: VIsion and Crowdsensing Technology for an Optimal Response in physical-securitY. http://madeirasic.us.es/victory/.

https://doi.org/10.1016/j.neunet.2020.09.013
0893-6080/

Nevertheless, when this is not enough, more data is needed. For this reason, we propose the use of synthetic data to enrich the datasets with realistic synthetic images. Therefore, using a video game engine, it is possible to simulate a realistic environment, adding characters and objects that are wanted to detect and cameras that circulate through the scenes to capture the images to include in the dataset.

Given these two problems, small objects and reduced number of labeled weapons in real environments, this paper applies an architecture based on the Feature Pyramid Network (FPN) and enhance the dataset with synthetic data, improving the detection of small objects and increasing the amount of data. This implementation is tested on a real dataset recorded by a CCTV and collected by the authors during a mock attack on the University of Seville[2] and on a dataset from another related study, analyzing how the use of this architecture and the synthetic data affects the results.

The goals of this study are: to improve the accuracy of object detection models using synthetic datasets, improve the current state-of-the-art in weapon detection using a FPN architecture with combinations of synthetic and real datasets and perform an analysis of a weapon detector on a real CCTV dataset, as these studies usually perform evaluations on unrealistic datasets.

Therefore, following these goals, this study presents three main points: (1) the release of two new weapons detection datasets, one obtained on real recordings with an infrastructure of security cameras used during a mock attack at a university, and another one that has been synthetically produced by using the Unity (2020) game engine; (2) an analysis of the behavior in object detection through the addition of synthetic datasets to real datasets in different proportions; (3) a number of models that improve the state-of-the-art of weapon detection.

The rest of the paper is organized as follows. In Section 2, related works in object detection in security environments, small object detection, and use of synthetic datasets are presented. Section 3 describes the datasets, the detection network used in our detection experiments, the models and the experimental setup that was used. Experiments and discussion of results are shown in Section 4. Finally, conclusions are drawn in Section 5.

2. Related work

2.1. Object detection in security monitoring environments

The detection of dangerous objects in CCTV is a difficult problem to solve. Nowadays, the security based on CCTV requires at least one person to be constantly monitoring everything that happens on one or more monitors simultaneously. This sometimes leads to overlooked threats and a delayed response to dangers. According to Velastin, Boghossian, and Vicencio-Silva (2006) typically after 20 min of CCTV monitoring, operators often fail to detect objects in a video scene. Ainsworth (2002) goes deeper into the analysis: after 12 min of continuous video monitoring, an operator is likely to miss up to 45% of screen activity and after 22 min of viewing, up to 95% of activity is overlooked. Therefore, combining human attention with a real-time detection system for dangerous objects can be highly valuable.

Among the proposed solutions, sliding windows with extractors features are the most commonly used (Grega, Matiolanski, Guzik, & Leszczuk, 2016). Nevertheless, this process is time-consuming, since it requires classifying each frame completely. Buckchash and Raman (2017) accelerate the process by using foreground segmentation Features from Accelerated Segment Test (FAST) based on feature detection for image localization, and Multi-Resolution Analysis (MRA) for classification and target confirmation, although it was tested just for detecting the shape of a knife.

Nonetheless, the most recent studies usually make use of Convolution Neural Network (CNN) trained with a set of images to detect weapons without having to completely scan the image, and are consequently faster and more suitable for detecting weapons in CCTV.

In 2017, Kibria and Hasan (2017) performed an analysis with the four algorithms they considered most relevant to identify knifes: Bag of Words (BoW) with linear Support Vector Machine (SVM), as classifier, and Speeded Up Robust Features (SURF), as feature descriptor; Histogram of Oriented Gradients (HOG) with SVM, to extract features and perform classification respectively; CNN; and Alexnet, as feature extractor, with SVM, as classifier. Among these four algorithms, the one that gets the best accuracy is CNN.

Olmos, Tabik, and Herrera (2018) made a comparison between the use of a CNN classifier over sliding windows and over Region Proposal Networks (RPN), the latter providing the best result, to detect handguns in video frames. Romero and Salamea (Romero & Salamea, 2019) proposed to carry out the detection in two steps, first detecting people using YOLO (Redmon, Divvala, Girshick, & Farhadi, 2015), and then detecting handguns on them with a CNN. Other studies use different techniques such as: applying preprocessing by adjusting the brightness to improve F1 by 12.24% over the detection of the most used types of cold steel weapons in crimes, as kitchen knife, machete, razor, dagger and carved knives, by Castillo, Tabik, Perez, Olmos, and Herrera (2019); minimizing the number of false positives at handgun detection by using a fusion of binocular images, by Olmos, Tabik, Lamas, Perez-Hernandez, and Herrera (2019); or using a binarization technique with One-Versus-All and One-Versus-One with small objects that are usually confused with a handgun or a knife when manipulated with hand: pistol, knife, smartphone, bill, purse and card, by Pérez-Hernández, et al. (2020).

2.2. Small object detection

Object detection has improved considerably in recent years. However, the detection of small objects remains a problem to be solved, presenting in the best dataset of the COCO'17 competition[3] an average precision (AP) of 0.345 for small objects, compared to 0.556 and 0.649 for medium and large objects respectively. Small objects obtain a lower precision in detection (38% less than medium and 47% less than large objects), and this is due to several reasons: (1) there are fewer annotations of small objects because not all the objects in the scene were (manually) annotated, as pointed out in Kisantal et al. (2019), although new datasets, such as Open Images (Kuznetsova, et al., 2020), mitigate this issue where 43% of the bounding boxes of their images occupy less than 1% of the image area; (2) these objects contain less information since they are smaller and convolutional networks do not identify them; (3) noise from compression or movement and occlusions affect small objects more negatively, as they present less information in the image, Ravichandran and Yegnanarayana (Ravichandran & Yegnanarayana, 1995) analyze the effect of noise on images in object recognition; (4) they represent a very small piece of the scene, which means that regions generated in object detection have higher probability of identifying medium or large objects.

[2] https://github.com/jossalgon/US-Real-time-gun-detection-in-CCTV-An-open-problem-dataset.

[3] COCO Detection Leaderboard. http://cocodataset.org/#detection-leaderboard.

When evaluating object detector models in general (Huang, et al., 2017) or in a specific domain such as traffic signs Arcos-Garcia, Alvarez-Garcia, and Soria-Morillo (2018) and Arcos-García, Álvarez-García, and Soria-Morillo (2018), some models achieve a better performance detecting small objects, such as Faster R-CNN with different backbones. However, most studies focus on improvements applied to specific problems and involve modifying current architectures, some of these are: the detection of traffic signs, improved with the proposal of Li, et al. (2017), which consists of applying a generative adversarial network to detect signals at a greater distance, or the modification of Faster R-CNN by Cao, et al. (2019); detection of objects with aerial views, where proposals such as the ones from Etten (2018) or Sommer et al. (2017) improve the location of objects in aerial images using new architectures optimized for this purpose; and remote sensing images, where Zhang, Wang, Thachan, Chen, and Qian (2018) propose an architecture with a deconvolution layer after the last convolution layer of the base network to improve the detection of small objects in remote sensing images.

On the other hand, there are some more general studies, among others: Lin, et al. (2017), who proposed FPN, a feature extractor that replaces the feature extractors used in object detectors (e.g. Faster R-CNN) to generate feature maps with higher quality information by using a structure in two pathways; Li, Ding, and Wang (2018) who propose the use of SSD (Liu, et al., 2016) replacing the original adding convolutional layers with dense skip connections to expand the receptive field of the whole network, thus improving performance by detecting small objects; Chen, Liu, Tuzel, and Xiao (2017) and Bosquet, Mucientes, and Brea (2018), which propose the use of a context model to choose the most promising regions in small objects, respectively applied to common objects and aerial views; Li and Yang (2018) modify the architecture of YOLOv2 to improve performance on small objects by adding changes such as a FPN, Liang, Shao, Zhang, and Gao (2018) make a similar modification using Faster R-CNN; Guan and Zhu (2017) propose a unified deep neural network building a Dense Faster R-CNN with an Atrous Region Proposal Network that explores object contexts at various scales sliding a set of atrous filters to which it is applied an increase of dilation rates in the last convolutional feature map.

Aforementioned, even though Open Images dataset already includes more small objects, their low occurrence is still a problem since it leads to training with only a few instances of them. Kisantal et al. (2019) propose to increase the number of instances of small objects by replicating them in the images, achieving a 7.1% improvement using unmodified Mask-RCNN architecture. In our approach we will also focus on increasing the apparitions of small objects to improve their detection precision, but by generating synthetically new scenes instead of replicating objects in the same image, since weapons do not usually appear in any area of the image.

2.3. Synthetic datasets

The use of synthetic data to augment or even generate new data is not a novel technique. In 2007, in the context of video surveillance, Taylor, Chosak, and Brewer (2007) proposed a virtual simulation based on Half-Life 2 game from Valve Software[4] for design and evaluate their system. In 2010, Marin, Vázquez, Gerónimo, and López (2010) use the same game to demonstrate that a model (linear Support Vector Machine using histogram of oriented gradients) can detect pedestrians in real images using virtual scenarios. More recently, Gaidon, Wang, Cabon, and Vig (2016) generated the vehicle video dataset "Virtual KITTI" using the game engine Unity,[5] and providing experimental evidence that pre-training models on real data behave similarly in real and virtual worlds, and pre-training on virtual data improves performance. Richter, Vineet, Roth, and Koltun (2016) also make use of synthetic datasets to compensate for the lack of real data, using "Grand Theft Auto V" game, for the semantic segmentation process in KITTI (Geiger, Lenz, Stiller, & Urtasun, 2013) and CamVid (Brostow, Fauqueur, & Cipolla, 2009) datasets. Combining synthetic and real images they improve the state-of-the-art concerning using only real images in KITTI and in CamVid. In 2018, Tremblay, et al. (2018), as part of the NVIDIA group, introduced the use of synthetic images with domain randomization (DR) for neural network training. They showed that by randomly perturbing the synthetic images during training, DR intentionally abandons photo-realism to force the network to learn the most important characteristics of the object to be detected, ignoring possible distractions in an image. This technique was introduced by Tobin, et al. (2017). Although the synthetic dataset is less realistic than the one found in Virtual KITTI, due to the DR, it presents slightly better results when using 1,000 or more real images in the fine-tuning process. Later on, Prakash, et al. (2019) improved the results using Structured DR, preserving some of the contexts, specifically the detection of small objects.

In 2019, Öhman (2019), used a military simulator called VBS3[6] to generate synthetic datasets by applying data augmentation (people with all types of weapons) on his training set. He used 34,000 images from the real-image dataset of Internet Movie Firearms Database (IMFDB).[7] Images were scraped from the website and manually labeled by The Swedish Defence Research Agency (not publicly available). The main objective was to check the best combination of real and synthetic images to detect weapons in the evaluation set: 15% of the IMFDB dataset (5135 images). Results obtained by Öhman using Faster R-CNN Inception V2 were not conclusive: synthetic data was useful to improve the results with only non realistic images. However, he combined the training datasets, not following the Tremblay proposal (Tremblay, et al., 2018): training first with synthetic and fine-tuning with real. Finally, the same year, the company Edgecase.ai,[8] released a synthetic gun detection dataset (World's First Synthetic Gun Detection Dataset from Edgecase.ai, 2020) following all the recommendations that came from DR (Tremblay, et al., 2018) and Structured DR (Prakash, et al., 2019) techniques. They use distractors as well as random textures (real and synthetic) over the generated scenes to enforce the network to pay attention to targeted objects like guns.

Our work evaluates previous weapons detection methods (Section 2.1) using the best techniques to detect small objects (Section 2.2) on existing datasets and tests the idea of using synthetic datasets (Section 2.3) to improve the learning task and the results. Since, as far as we know, there is no realistic CCTV weapon image data set, we include a labeled one and provide a baseline detection result that improves the state of the art.

3. Datasets and detection network

This section describes the datasets used for training and testing the object detector, as well as the detection network used during the experimentation.

[4] Valve Software Corporation. http://www.valvesoftware.com.

[5] Unity Game Engine. https://unity.com/.

[6] Virtual Desktop Training & Simulation Host. https://bisimulations.com/products/vbs3.

[7] Internet Movie Firearms Database. http://www.imfdb.org/wiki/Main_Page.

[8] Edgecase - https://www.edgecase.ai/.

Table 1
Details of datasets used in this study with acronym, name, number of images and type of annotations.

Acronym	Dataset	N. Images	Annotation
G	UGR - Handgun dataset for the region proposals approach (Split1)	2700	Detection
M	US - Mock Attack Cam 1, 7	4118	Detection
E	Edgecase - Gun Detection Synthetic Data	1983	Detection
U0.5/U1/U2.5	US -Unity Synthetic Dataset	500/1000/2500	Detection
Testset1	UGR - Handgun testset	608	Classification
Testset2	UGR - Handgun dataset for the region proposals approach (Split2)	300	Detection
Testset3	US - Mock Attack Cam 5	1031	Detection
Testset4	US - Normal Cam 5	20,260	No weapons

Table 2
Bounding box sizes of each camera.

Class	Small boxes			Medium boxes			Large boxes			All boxes			
	C1	C5	C7	C1	C5	C7	C1	C5	C7	C1	C5	C7	Total
Handgun	94	67	104	160	1029	243	6	8	3	260	1104	350	1714
Rifle	15	5	13	184	353	134	21	51	21	220	409	168	797
All	109	72	117	344	1382	377	27	59	24	480	1513	518	2511

C1, C5 and C7 stands for Camera 1, 5 and 7.

3.1. Datasets

In order to build a more complete dataset to achieve our goals, we have compiled a number of previously existing datasets, a manual annotated dataset generated from mock attack videos recorded for this purpose and a set of synthetic images generated by modeling virtual scenarios, also created for this purpose, with automatic annotations using a video game engine. The datasets are presented in Table 1 that shows the dataset name, the number of images it contains, the code that will identify each one later and the type of annotations it presents, being these, "detection" if it contains bounding boxes, "classification" if it does not contain bounding boxes but the class and "no weapons" if it does not contain objects to detect.

The datasets G, Testset1 and Testset2 were published by the University of Granada and used in the study of Olmos et al. (2018). Originally G and Testset2 were combined in the same dataset but, in order to perform a detection evaluation, the original dataset was split into two different partitions (90% for train and 10% for test), as Edgecase (World's First Synthetic Gun Detection Dataset from Edgecase.ai, 2020) does.

The images from dataset M (cameras 1 and 7) were collected, annotated and used as train set, while camera 5 was used as test set (Testset3), since it is an important camera to be tested as it is located at the entrance of a university module. All the people that appear in Testset3 have given and signed their consent to appear in the dataset. Another dataset was also generated with camera 5 during a normal day (Testset4), without the execution of any mock attack or presence of weapons, which allows us to analyze the false positives produced by the detector. Images shown in the paper from Testset4 have been anonymized and will not be released in order to preserve the privacy of the people who appear in it.

The generated datasets "Mock attack dataset" (M and Testset3) and "Unity synthetic dataset" (U0.5, U1 and U2.5) are described in more detail below.

3.1.1. Mock attack dataset

This dataset has been manually annotated and collected during a mock attack, after obtaining all the permissions by our University and the security personnel. Details are presented below, indicating each of the cameras used during the mock attack and the scenarios they present.

Infrastructure for data acquisition is composed of three surveillance cameras located at different places in the same area covering two different corridors and one entrance, forming different scenarios. The training set is composed of images from cameras 1 and 7 (Dataset M), while the test set by images from camera 5 (Testset3). The description of each camera is as follows:

- Cam1: located in one of the two corridors, it presents some conflicting objects, such as doors or bins, and the lighting is uniform. The time of the video sequence for this camera is 40 min and 25 s. Selecting the five segments with movement, the total duration is 5 min and 4 s. These segments were manually annotated at 2 frames per second (FPS), resulting in a total of 607 frames.
- Cam7: located in the other corridor, this camera presents similarities with Cam1, both in scenery and lighting. However, Cam7 presents more conflicting objects, such as a fire extinguisher or objects on the walls. The duration of the sequence for this camera is 1 h, 3 min and 34 s. Selecting the segments with movement, the total duration is 29 min and 16 s. These segments were annotated at 2 FPS, resulting in a total of 3511 frames.
- Cam5: located at the entrance of a university module, presents some conflicting objects, such as a black carpet on the floor, and also irregular lighting with rays covering part of the scene. The duration of the video sequence for this camera is 39 min and 7 s. Choosing segments with movement, the time decreases to 8 min and 36 s. This sequence was annotated at 2 FPS, resulting in a total of 1031 frames.

Fig. 1 shows a frame of each camera and Table 2 presents the labeled bounding boxes of each one. Size of boxes are considered following COCO metrics:[9] Small: area $< 32\ pixels^2$, Medium: $32\ pixels^2 <$ area $< 96\ pixels^2$ and Large: area $> 96\ pixels^2$.

3.1.2. Unity synthetic dataset

This dataset was generated by modeling in Unity Game Engine a scenario that emulates a part of a city and an educational center within it (Fig. 2, up left and right). Several cameras capture the movements of multiple characters, made up of 11 different models and 7 animations. These images enhance the generated datasets with 11 different objects: 4 types of handguns, 5 types of rifles, a knife, and a smartphone. This dataset is not photo-realistic since, as we indicated in Sub Section 2.3, the generation of unrealistic data can help the network to focus on the object to be detected. To capture the images and create the annotations, cameras are first placed in the scenes and the characters are

[9] http://cocodataset.org/#detection-eval.

Fig. 1. Description of datasets - Frames of Camera 1 (up left), Camera 7 (up right) and Camera 5 (down).

assigned to follow a path through the scenarios. These characters present a random component that allows them to turn around or wait a certain time in "waypoints". Every time the simulation is executed, different datasets are generated.

While the modeled actors walk through the scenario, the system automatically captures images from the different cameras and annotate them. For this purpose, actors include a script to obtain the character's "GameObject". Cameras also include another script that throws rays over the annotated objects. These rays identify if the object is visible or hidden (partially or totally). With these scripts, the algorithm detects if the object is visible in at least 70% of its totality. If it fits this criterion, maximum and minimum coordinates are obtained with the camera image as a reference. Finally, the algorithm writes these data in XML files in PASCAL VOC format, including its class and image information.

3.2. Detection network

As it has been stated in Sub Section 2.2, Faster R-CNN behaves better for small objects (Arcos-Garcia et al., 2018; Huang, et al., 2017). Furthermore, placing FPN (Lin, et al., 2017) in Faster R-CNN, the performance on small objects is increased by 12.9 points.

Considering that the object type to detect in this study usually has small sizes compared to the whole image, we have considered more appropriate to use implementations with FPN features and a model that balances precision and inference speed, which is necessary in order to implement it on real-time CCTV, able to report in the lowest possible time to the security guard.

To train this architecture, we first used Fine-tuning with the frozen weights of the COCO dataset, one of the largest and most tested datasets to date, to take advantage of previous training on a large number of images.

The object detection models used in this study were trained on the detectron2 framework (Wu, Kirillov, Massa, Lo, & Girshick, 2019) with the Faster R-CNN-FPN architecture with the resnet50 backbone. This configuration obtains an mAP of 40.2[10] over the COCO dataset, with an execution time of 90 ms per image of our CCTV dataset using an NVIDIA GeForce GTX-1080Ti card, providing a good balance between mAP and execution time.

3.3. Experimental setup

To achieve the goals exposed in the introduction of this study, the model based on Faster R-CNN-FPN architecture with ResNet-50 has been trained with 14 different dataset combinations and tested with four datasets as described in Table 3. All the experiments were executed in a GeForce GTX-1080Ti with 11 GB of memory.

The datasets used during the training were augmented with a horizontal random flip, adding more data while preserving the

[10] https://github.com/facebookresearch/detectron2/blob/master/MODEL_ZOO.md.

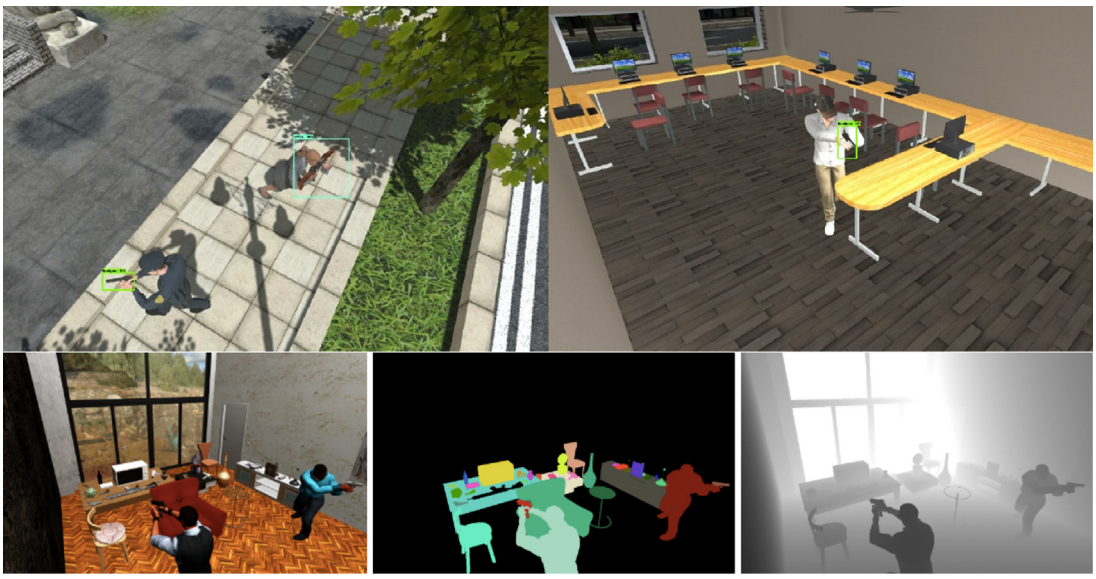

Fig. 2. Synthetic data generation: Unity dataset (ours) Outdoors (up left) and indoors (up right) scenarios. Edgecase scenarios (down).

Table 3
Training sets used to train the 14 models and testing datasets where the models are evaluated.

Training sets	Type
E U0.5 U1 U2.5	Only Synthetic
G M G + M	Only Real
E + G U0.5 + G U1 + G U2.5 + G E + G + M U0.5 + G + M U2.5 + G + M	Synthetic + Real
Testing sets	**Type**
Testset1	Non realistic
Testset2	Non realistic: only some CCTV
Testset3	Realistic: only CCTV
Testset4	Realistic: only CCTV (non weapon)

coherence of the image. Models with only real images or only synthetic images are trained for 80k steps. Models using both datasets are pre-trained first using the synthetic dataset during 40k steps and then fine-tuned with real images for additional 40k steps, as shown in Fig. 3.

Only the class "weapon" was used to detect handguns and rifles, since the main objective is to detect dangerous objects, and we are not interested in differentiating them, knives and other weapons were not included in this study, but they will be studied as future work. Anchors of different sizes and scales were used to detect weapons within the image. The choice of different sizes and scales of anchors allows the detector to find objects of specific dimensions and shapes. After analyzing the class "weapon", we observed that sizes 32 and 64 $pixels^2$ allow us to detect guns, since they usually present that size in the image; meanwhile, sizes 128, 256 and 512 $pixels^2$ allow us to detect rifles, or guns in the foreground, since, as the previous case, they usually present this size in the image. These anchors also present three different ratios to cover the different shapes of the object, these ratios are "1:2" and "2:1", since it is the ratio that usually presents the rifles, and "1:1", since it is the ratio that usually presents the guns and in some cases the rifles. These four scales and three ratios cover all the most common possibilities in which these objects appear. Other scales and ratios were also studied, however, this configuration was the one with the best results due to the nature of the desired objects.

The models were trained with a batch size of 2 and a learning rate of $2 \cdot 10^{-3}$ in the first 40k steps, and then $2 \cdot 10^{-4}$ up to 80k steps, as this allowed us to avoid over-fitting and obtain better results.

Combinations with synthetic data are disjointed, so when "Edgecase dataset (E)" is applied, "Unity dataset (U)" will not be used, and vice versa, to study which one performs better.

The following datasets were used during experiments evaluation: (1) "Testset1", composed of foreground images and no annotated, to compare the results with the ones in the study of Olmos et al. (2018); (2) "Testset2", to analyze how the experiments perform against weapons images at a medium and short distance; (3) "Testset3", to analyze the in the context of a real CCTV, with far weapons images; (4) "Testset4", using sequences where no weapon was present to analyze how the experiment performs in the case of no danger (false positives).

4. Experiments and discussion of results

As mentioned in Sub Section 3.3, 14 models with different combinations were performed over four test datasets. This variety of combinations allowed us to understand how the model performs with different training data and to analyze how the use of synthetic data impacts on it; also, the four test datasets allowed us to analyze and compare the performance of the different models in different contexts. Although the test set and the training set usually must satisfy the condition of the same distribution, in our case it is impossible to include synthetic images to the test set since they represent a real situation

The results were obtained using the following detection COCO metrics[11] and other standard metrics: Average Precision (AP), used as COCO primary challenge metric; AP 0.5, used also as PASCAL VOC metric; AP 0.75, strict metric; AP Across Scales, being this, APs, APm and APl; and True Positives (TP), False Positives (FP), False Negatives (FN), Precision, Recall, and F1-score,

[11] http://cocodataset.org/#detection-eval.

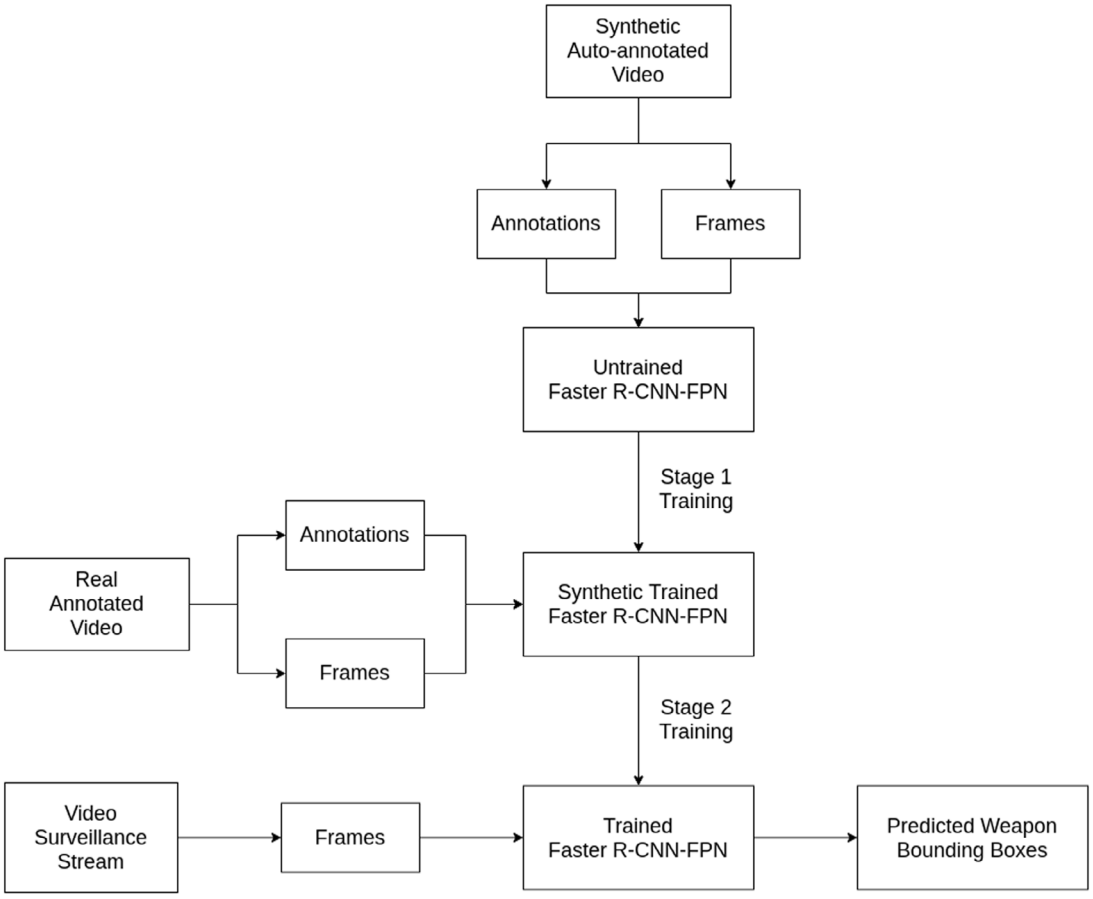

Fig. 3. Block scheme of the complete procedure, including two stage training with synthetic and real images, and the use of the trained model.

obtained with an IoU of 0.50 and a confidence limit of 0.95 or 0.99.

4.1. Experiment 1

The first evaluation uses the dataset "Testset1". Fig. 4 shows some images of this dataset. This test dataset was chosen to analyze how the different combinations perform against images of weapons at a short distance and to compare the results with those obtained by Olmos et al. (2018).

As shown in Table 4, models trained with only real images (*G*, *M*, and *G+M*), achieve high accuracy and recall only when using the *G* dataset, which contains images of weapons at short and medium distance (similar to the test dataset). The model trained only with dataset *M*, is not useful at all (precision of 9.09% and a recall of 0.33%). The same occurs with the models trained with synthetic images (*E*, *U0.5*, *U1*, and *U2.5*). These results can be explained because the simulations are based on weapons images at medium or long distances, while test datasets contain weapons at a short distance (Fig. 4).

Best results are obtained with models trained combining synthetic and real images, specifically *E+G* and *U2.5+G+M*, improving the state-of-the-art from the work by Olmos et al.

In this experiment the metrics Precision, Recall and F1 were used since the original evaluation is based on a classification model (without IoU). In the following experiments complete detection models are evaluated, so average accuracy (AP) is used as the main metric, instead Precision, Recall and F1, as usual in works related to deep learning for detection.

Table 4
Results of dataset combinations over "UGR - Handgun testset" (Testset1). The metrics used were True Positives (TP), False Positives (FP), False Negatives (FN), Precision (Prec), Recall (Rec) and F1 score. IoU used of 0.5 and confidence threshold of 0.95. Nonetheless, Olmos et al. (2018) do not specify the confidence threshold used.

Model	TP	FP	FN	Prec	Rec	F1
E	1	1	303	50.00%	0.33%	0.65%
U0.5	0	1	304	0.00%	0.00%	0.00%
U1	0	1	304	0.00%	0.00%	0.00%
U2.5	0	0	304	0.00%	0.00%	0.00%
G	302	54	2	84.83%	99.34%	91.52%
M	1	10	303	9.09%	0.33%	0.63%
G + M	304	52	0	85.39%	100.00%	92.12%
E + G	**304**	**41**	**0**	**88.12%**	**100.00%**	**93.68%**
U0.5 + G	304	44	0	87.36%	100.00%	93.25%
U1 + G	303	42	1	87.83%	99.67%	93.37%
U2.5 + G	303	44	1	87.32%	99.67%	93.09%
E + G + M	303	44	1	87.32%	99.67%	93.09%
U0.5 + G + M	303	40	1	88.34%	99.67%	93.66%
U2.5 + G + M	**304**	**41**	**0**	**88.12%**	**100.00%**	**93.68%**
Olmos et al. (2018)	304	57	0	84.21%	100.00%	91.43%

4.2. Experiment 2

The second evaluation uses the dataset "Testset2". Fig. 5 shows some images from this dataset. These images were selected to compare the results with those obtained by Edgecase (World's First Synthetic Gun Detection Dataset from Edgecase.ai, 2020) (they only point out AP50 and AR) and to analyze how the different combinations perform against images of weapons at a

Fig. 4. Some images from "Testset1".

Fig. 5. Some images from "Testset2".

medium or short distance. In order to minimize the false positives, a confidence threshold of 0.99 was considered in metrics TP, FP and FN.

Table 5 shows that models trained only with synthetic images (*E*, *U0.5*, *U1*, and *U2.5*) perform extremely bad. The same with the model trained with real images from CCTV (*M*). Most of the images of this dataset are in the foreground, which extremely differs from synthetic images and real ones from CCTV. Only when training with *G* dataset, the results improve (the training set is very similar to the testing one).

In this test dataset, the best result, considering it the one with the highest AP, is obtained combining synthetic and real datasets (*U0.5 + G*), that is: fine-tuning first with 500 images generated with the Unity video game engine and then fine-tuning with images of weapons at short and medium distances, since they have a similar context. Specifically, it achieves the best AP and AP50, 3.9 points better than Edgecase in AP50.

According to Table 6, adding synthetic datasets to the dataset *G* results on different improvements and deterioration in the AP by object size depending on the specific dataset used and its size. When using the synthetic dataset *E*, results show an improvement in large objects of 0.90 (a 4.59% increase), and 0.10 (a 3.03% increase) in medium objects, however, no improvement in small objects can be found. On the other hand, applying the Unity dataset *U* with different proportions shows the following: with 500 images, the AP of small objects is improved by 0.80 points, whereas it is degraded in medium and large objects by 0.50 and 0.20 points respectively (a 15.15% and 1.02% decrease, respectively); and with 2,500 images there is an improvement of 0.20 points in small objects, a degradation of 0.60 (a 18.18% decrease) in medium objects and an improvement of 2.40 (a 12.24% increase) in large objects. Hence, the greatest improvement for small objects is obtained when using 500 images (dataset *U0.5*), and for large objects when using 2,500 images (dataset *U2.5*).

Table 5
Results of dataset combinations over "UGR - Handgun dataset for the region proposals approach" (Testset2). The metrics used were Average Precision (AP), AP50, AP75, AP Across Scales (APs, APm and APl), and True Positives (TP), False Positives (FP) and False Negatives (FN) with an IoU of 0.5 and confidence threshold of 0.99.

							IoU 0.5, C > 0.99		
Model	AP	AP50	AP75	APs	APm	APl	TP	FP	FN
E	4.9	10.2	4.2	0.3	2.1	9.4	15	17	322
U0.5	3.8	11.6	1.0	7.1	5.2	3.5	18	7	319
U1	5.3	14.6	2.0	6.4	8.3	5.0	16	7	321
U2.5	3.1	8.1	2.4	4.8	5.7	2.9	7	3	330
G	65.2	88.1	71.3	28.6	47.8	71.0	271	13	66
M	0.9	3.4	0.3	1.5	2.5	2.3	4	6	333
G + M	65.9	90.7	71.4	30.3	49.9	71.2	275	13	62
E + G	66.9	90.7	71.3	**33.7**	48.3	72.3	261	6	76
U0.5 + G	**68.3**	**92.6**	**74.1**	29.0	50.2	**74.2**	261	6	76
U1 + G	67.9	90.9	73.0	28.9	**51.8**	73.3	260	5	77
U2.5 + G	67.2	91.6	71.6	30.5	47.2	73.0	265	6	72
E + G + M	66.5	91.4	69.1	32.0	49.5	71.9	256	7	81
U0.5 + G + M	68.2	91.7	73.8	30.5	49.6	73.8	259	5	78
U2.5 + G + M	67.7	91.6	72.7	31.8	50.3	73.1	253	2	84
Edgecase (World's First Synthetic Gun Detection Dataset from Edgecase.ai, 2020)	–	88.7	–	–	–	–	–	–	–

Table 6
Increase of points in AP by size of bounding box (s, m and l stands for small, medium and large) in relation to the model of the dataset *G* and the use of synthetic datasets.

Model	APs	APm	APl	APs Incr.	APm Incr.	APl Incr.
G	0.00	3.30	19.60	–	–	–
E + G	0.00	3.40	20.50	0.00	0.10	0.90
U0.5 + G	0.80	2.80	19.40	0.80	−0.50	−0.20
U2.5 + G	0.20	2.70	22.00	0.20	−0.60	2.40

Table 7
Results of dataset combinations over "US - Mock Attack Cam 5" (Testset3). The metrics used were Average Precision (AP), AP50, AP75, AP Across Scales (APs, APm and APl), and True Positives (TP), False Positives (FP) and False Negatives (FN) with an IoU of 0.5 and confidence threshold of 0.99.

							IoU 0.5, C > 0.99		
Model	AP	AP50	AP75	APs	APm	APl	TP	FP	FN
E	0.1	0.3	0.0	0.0	0.2	1.0	3	24	1510
U0.5	0.7	1.2	1.0	0.0	0.3	3.0	8	20	1505
U1	0.7	1.0	1.0	0.0	0.1	2.4	5	48	1508
U2.5	0.7	1.2	1.0	0.0	0.4	2.4	2	3	1511
G	3.9	11.2	1.7	0.0	3.3	19.6	82	46	1431
M	12.7	35.9	5.1	2.6	12.8	25.5	293	53	1220
G + M	12.8	36.6	5.0	1.9	12.6	26.7	309	56	1204
E + G	3.9	10.6	1.8	0.0	3.4	20.5	48	5	1465
U0.5 + G	3.4	10.5	1.8	0.8	2.8	19.4	68	17	1445
U1 + G	3.8	10.8	1.7	0.6	2.8	22.3	52	17	1461
U2.5 + G	3.5	9.6	2.0	0.2	2.7	22.0	55	20	1458
E + G + M	**14.6**	**40.8**	**6.8**	2.7	**14.7**	**33.9**	250	25	1263
U0.5 + G + M	13.8	37.4	6.6	**3.6**	13.6	32.0	209	24	1304
U2.5 + G + M	13.3	35.5	6.3	2.3	13.0	33.2	209	27	1304

4.3. Experiment 3

The third evaluation uses the dataset "Testset3". This dataset was selected to analyze how the combinations perform in the context of a real CCTV, with images of weapons at long distance.

Table 7 shows that the model trained with a combination of images from the same context (M), such as *G+M*, *E+G+M*, *U0.5+G+M* and *U2.5+G+M* improve the AP and AP50 to the rest of the models trained only with real data. Also, as in the previous experiments, models trained only with synthetic images (*E*, *U0.5*, *U1*, and *U2.5*) performs extremely bad.

Thus, the best solution to achieve a high AP is to combine both real datasets with synthetic data (*E+G+M*), that is: training with images at short-medium (*G*) and large distances (*M*) and performing a previous training with synthetic data (*E*).

Table 8
Increase of points in AP by size of bounding box in relation to the model of the dataset *G + M* and the use of synthetic datasets.

Model	APs	APm	APl	APs Incr.	APm Incr.	APl Incr.
G + M	1.90	12.60	26.70	–	–	–
E + G + M	2.70	14.70	33.90	0.80	2.10	7.20
U0.5 + G + M	3.60	13.60	32.00	1.70	1.00	5.30
U2.5 + G + M	2.30	13.00	33.20	0.40	0.40	6.50

According to Table 8, when synthetic datasets are considered together with the *G + M* combination other improvements in the AP by size are achieved, with no loss in any case. Using the synthetic dataset *E* shows an increment of 0.80 points (a 42.11% increase) in small objects, 2.10 (a 16.67% increase) in medium objects and 7.20 (a 26.97% increase) in large objects. However, when using 500 images from Unity (dataset *U0.5*), although the improvement in medium and large objects is not high, being 1.0 and 5.30 respectively (a 7.94% and 19.85% increase, respectively), it is better in small objects, being 1.70 points (a 89.47% increase). Using 2,500 images from Unity (dataset *U2.5*) produces a lower increase, resulting in an improvement of 0.40 points on small objects (a 21.05% increase), 0.40 on medium objects (a 3.17% increase) and 6.50 on large objects (a 24.34% increase). Therefore, the greatest improvement for small objects is obtained when adding 500 Unity images (dataset *U0.5*), as it improves 89.47% in AP with the use of synthetic dataset. Although we can also see improvement in large objects when using dataset *E*, which obtain a greater improvement.

Two images from this evaluation can be seen in Fig. 6.

4.4. Experiment 4

The fourth evaluation uses the dataset "Testset4". This dataset was used to measure the false positives that can appear without using pre-processing alerts. This dataset, as it is collected with the same camera, contains the same shot as "Testset3", and therefore, the objects presented in the scene are far from the camera.

Table 9 shows that the dataset *U2.5*, which uses only 2500 Unity synthetic images, has the least false positives rate, obtaining only 26 FP. However, as already seen in the results of Table 7, it also obtains worse AP, with a high FN (IoU 0.5 and confidence $>$ 0.99), which indicates that it detects very few objects, so it will get it low instances wrong yet it will also get it low instances right. The same applies to the second best,

Fig. 6. Images of third experiment, using "Testset3".

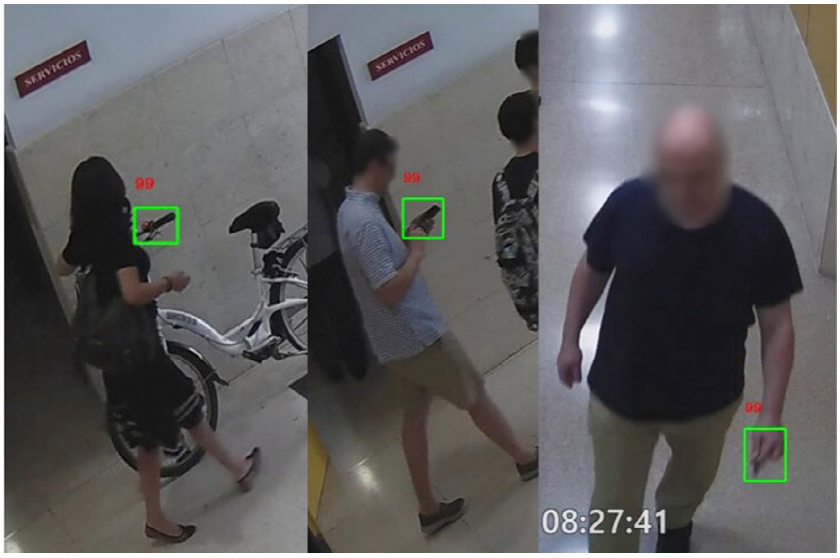

Fig. 7. False positives found on "Testset4", with no weapon sequences and conflicting objects (e.g. bicycle, mobile or keys).

Table 9
Results of dataset combinations over "US - Normal Cam 5" (Testset4). False positives (FP), with a confidence threshold of 0.99, using Cam 5 during a normal day with no weapons.

Model	FP(0.99)
E	241
U0.5	821
U1	1062
U2.5	**26**
G	615
M	492
G+M	686
E + G	201
U0.5 + G	514
U1 + G	358
U2.5 + G	478
E + G + M	209
U0.5 + G + M	298
U2.5 + G + M	249

E+G. The model that obtained the best results in the third test dataset (Table 7), *E+G+M*, obtains only 209 FP here, it is the third experiment with lower false positives rates, and taking into account that the two best models here obtain low APs in the presence of weapons at long distances, we can conclude that this model (E+G+M) obtains a satisfactory result, presenting the highest AP with weapons and the lowest FP in this dataset.

Some false positives are shown in Fig. 7.

4.5. Discussion of results

From previous experiments, we can conclude that our proposal improves the state of the art in weapon detection (Olmos et al., 2018) with the same recall and an increase of 3.91% in precision and 2.25% in F1. Also, in the second experiment, we improve the results obtained by Edgecase (World's First Synthetic Gun Detection Dataset from Edgecase.ai, 2020), achieving 3.9 points better in AP50. In both cases, results show that the combination of first training in synthetic images and then training or fine-tuning in real images, is the best option, as well as improving

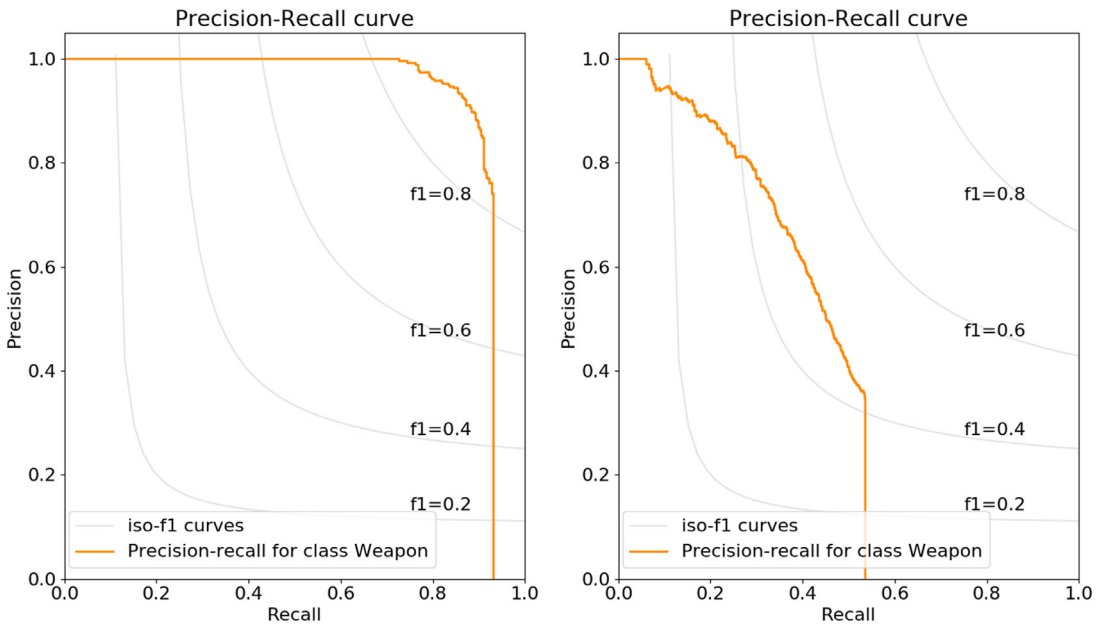

Fig. 8. Plot Precision–Recall curve and iso-f1 curves for evaluation on first experiment trained using *U2.5+G+M* (left) and third one trained using *E+G+M* (right).

the AP of small objects by 0.80 points. Although our synthetic images are less realistic than those from Edgecase dataset, both synthetic datasets influence positively in all the combinations. *E* and *U2.5* behave very well in the first experiment, *U0.5* a better in the second, and *E* better in experiment 3, although *U0.5* achieves an improvement of 1.70 points (a 89.47% increase) in AP of small objects.

As a consequence, we can observe that for the short and medium distance with non-realistic test datasets (test dataset 1 and 2), results improve training with synthetic data and then refining them with real data. This gets an object detector model that surpasses the current state-of-the-art, considering also the limitation when choosing the architecture to use, since the inference time should be low while preserving a high precision, obtaining an inference time of 90 ms (using an NVIDIA GeForce GTX-1080Ti card). Experiments 1 and 2 are far from real situations. Experiments 3 and 4 show that the detector, even using Faster R-CNN with FPN, has a very low AP for small objects (3.6 in the best case) and AP for medium ones (14.7), even for large (33.9). The reason for this low AP is due to the poor visibility caused by movement, the long distance to the weapons and therefore the size of the objects at the scene. Partial occlusions of weapons are also a key factor for these results. As shown in the precision–recall curve in Fig. 8, controlled experiments, such as the first and second ones, are far from real situations present in experiments 3 and 4.

5. Conclusions

There are many problems in bringing weapons detection to real CCTV scenarios. The distance from the object to the camera is critical, improving the detection as the object is closer to the camera.

In this work, we have presented the behavior of a Faster R-CNN object detector using FPN and trained using synthetic and real images in a real CCTV. This work improves the state-of-the-art in existing datasets and shows that training using synthetic data increase the accuracy by adding more variety to the model and real data improve the detection of CCTV images. However, results show that there is room for improvement in real scenarios, as low average precision is obtained and the inference time needs also to be improved to get closer to a real-time detection system. To partially fix this situation, we have published a new dataset based on a mock attack at a University, so that other researchers can try out new algorithms. We have also released another dataset created with Unity game engine for pre-train the models and increase their accuracy.

Because of the small size of weapons at that distance and the partial occlusions in many frames, weapon detection in real CCTV scenarios is a challenging task, resulting in an open problem. According to the extensive analysis carried out, we observe that the trained models can recognize the visual characteristics of the weapons, detecting most of them on the scene. However, in some frames and regardless of context, this task is difficult even for a human.

As research lines to be followed to increase the recall and accuracy, we consider that a combination of object detectors with action recognition (Wang, et al., 2016), 3D pose estimation (Dabral, et al., 2018), and violence detection (Gao, Liu, Sun, Wang, & Liu, 2016) could boost the metrics. To reduce the number of false positives we recommend following these alternatives: (1) using a specific classifier to reduce the detection of usual objects such as bicycles, mobile phones or keys, and (2) consider only consecutive frames with detected weapons to raise an alert.

We must remember that supervised detection systems are silenced on many occasions by the security guards due to the false positive rate that they provide. Minimizing these cases makes the difference between one functional system and another that is not at all. Following these proposals and the knowledge generated in this work, we hope to reduce the mass shootings' impact generating faster alarms and reducing the number of victims.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This research is partially supported by The Spanish Ministry of Economy and Competitiveness MINECO/FEDER R&D, UE through the project VICTORY (Grant No.: TIN2017-82113-C2-1-R).

References

Ainsworth, T. (2002). Buyer beware. *Security Oz*, *19*, 18–26.

Arcos-García, Á., Álvarez-García, J. A., & Soria-Morillo, L. M. (2018). Deep neural network for traffic sign recognition systems: An analysis of spatial transformers and stochastic optimisation methods. *Neural Networks*, *99*, 158–165. http://dx.doi.org/10.1016/j.neunet.2018.01.005, URL http://www.sciencedirect.com/science/article/pii/S0893608018300054.

Arcos-Garcia, A., Alvarez-Garcia, J. A., & Soria-Morillo, L. M. (2018). Evaluation of deep neural networks for traffic sign detection systems. *Neurocomputing*, *316*, 332–344.

Bosquet, B., Mucientes, M., & Brea, V. M. (2018). STDnet: A ConvNet for small target detection. In *BMVC*.

Brostow, G. J., Fauqueur, J., & Cipolla, R. (2009). Semantic object classes in video: A high-definition ground truth database. *Pattern Recognition Letters*, *30*(2), 88–97.

Buckchash, H., & Raman, B. (2017). A robust object detector: Application to detection of visual knives. In *IEEE international conference on multimedia and expo workshops. IEEE international conference on multimedia and expo workshops*.

Cao, C., Wang, B., Zhang, W., Zeng, X., Yan, X., Feng, Z., Liu, Y., & Wu, Z. (2019). An improved faster r-CNN for small object detection. *IEEE Access*, *7*, 106838–106846. http://dx.doi.org/10.1109/ACCESS.2019.2932731.

Castillo, A., Tabik, S., Perez, F., Olmos, R., & Herrera, F. (2019). Brightness guided preprocessing for automatic cold steel weapon detection in surveillance videos with deep learning. *Neurocomputing*, *330*, 151–161. http://dx.doi.org/10.1016/j.neucom.2018.10.076.

Chen, C., Liu, M.-Y., Tuzel, O., & Xiao, J. (2017). R-CNN for small object detection. In *Lecture Notes in Computer Science*: *vol. 10115*, *Computer vision - ACCV 2016*, *PT V* (pp. 214–230). http://dx.doi.org/10.1007/978-3-319-54193-8_14.

Dabral, R., Mundhada, A., Kusupati, U., Afaque, S., Sharma, A., & Jain, A. (2018). Learning 3d human pose from structure and motion. In *Proceedings of the European conference on computer vision* (pp. 668–683).

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition* (pp. 248–255). Ieee.

Enríquez, F., Soria, L. M., Álvarez-García, J. A., Caparrini, F. S., Velasco, F., Deniz, O., & Vallez, N. (2019). Vision and crowdsensing technology for an optimal response in physical-security. In *International conference on computational science* (pp. 15–26). Springer.

Etten, A. V. (2018). You only look twice: Rapid multi-scale object detection in satellite imagery. arXiv:1805.09512.

Gaidon, A., Wang, Q., Cabon, Y., & Vig, E. (2016). Virtual worlds as proxy for multi-object tracking analysis. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4340–4349).

Gao, Y., Liu, H., Sun, X., Wang, C., & Liu, Y. (2016). Violence detection using oriented violent flows. *Image and Vision Computing*, *48*, 37–41.

Geiger, A., Lenz, P., Stiller, C., & Urtasun, R. (2013). Vision meets robotics: The kitti dataset. *International Journal of Robotics Research*, *32*(11), 1231–1237.

Grega, M., Matiolanski, A., Guzik, P., & Leszczuk, M. (2016). Automated detection of firearms and knives in a CCTV image. *Sensors*, *16*(1), http://dx.doi.org/10.3390/s16010047.

Guan, T., & Zhu, H. (2017). Atrous faster r-CNN for small scale object detection. In *2017 2nd International conference on multimedia and image processing* (pp. 16–21). http://dx.doi.org/10.1109/ICMIP.2017.37.

Huang, J., Rathod, V., Sun, C., Zhu, M., Korattikara, A., Fathi, A., Fischer, I., Wojna, Z., Song, Y., Guadarrama, S., et al. (2017). Speed/accuracy trade-offs for modern convolutional object detectors. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 7310–7311).

Kibria, S. B., & Hasan, M. S. (2017). An analysis of feature extraction and classification algorithms for dangerous object detection. In *2017 2nd International conference on electrical & electronic engineering* (pp. 1–4). IEEE.

Kisantal, M., Wojna, Z., Murawski, J., Naruniec, J., & Cho, K. (2019). Augmentation for small object detection. arXiv:1902.07296.

Kuznetsova, A., Rom, H., Alldrin, N., Uijlings, J., Krasin, I., Pont-Tuset, J., Kamali, S., Popov, S., Malloci, M., Duerig, T., et al. (2020). The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. *International Journal of Computer Vision*.

Li, C., Ding, Y.-d., & Wang, D.-b. (2018). Dense-SSD for detecting small objects. In *2018 International conference on electrical, control, automation and robotics (vol. 307)* (pp. 486–492).

Li, J., Liang, X., Wei, Y., Xu, T., Feng, J., & Yan, S. (2017). Perceptual generative adversarial networks for small object detection. In *30th IEEE conference on computer vision and pattern recognition* (pp. 1951–1959). http://dx.doi.org/10.1109/CVPR.2017.211.

Li, R., & Yang, J. (2018). Improved YOLOv2 object detection model. In *Proceedings of 2018 6th international conference on multimedia computing and systems* (pp. 64–69).

Liang, Z., Shao, J., Zhang, D., & Gao, L. (2018). Small object detection using deep feature pyramid networks. In *Lecture notes in computer science*: *vol. 11166*, *Advances in multimedia information processing, PT III* (no. III), (pp. 554–564). http://dx.doi.org/10.1007/978-3-030-00764-5_51.

Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2117–2125).

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., & Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In *European conference on computer vision* (pp. 740–755). Springer.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. C. (2016). SSD: Single shot multibox detector. In *Lecture Notes in Computer Science*: *vol. 9905*, *Computer vision - ECCV 2016*, *PT I* (pp. 21–37). http://dx.doi.org/10.1007/978-3-319-46448-0_2.

Marin, J., Vázquez, D., Gerónimo, D., & López, A. M. (2010). Learning appearance in virtual scenarios for pedestrian detection. In *2010 IEEE computer society conference on computer vision and pattern recognition* (pp. 137–144). IEEE.

Miller, G. A. (1998). *WordNet: An electronic lexical database*. MIT Press.

Öhman, W. (2019). *TRITA-EECS-EX*, *Data augmentation using military simulators in deep learning object detection applications* (Master's thesis), (2019:638), (p. 62). KTH, School of Electrical Engineering and Computer Science (EECS).

Olmos, R., Tabik, S., & Herrera, F. (2018). Automatic handgun detection alarm in videos using deep learning. *Neurocomputing*, *275*, 66–72. http://dx.doi.org/10.1016/j.neucom.2017.05.012.

Olmos, R., Tabik, S., Lamas, A., Perez-Hernandez, F., & Herrera, F. (2019). A binocular image fusion approach for minimizing false positives in handgun detection with deep learning. *Information Fusion*, *49*, 271–280. http://dx.doi.org/10.1016/j.inffus.2018.11.015.

Pérez-Hernández, F., Tabik, S., Lamas, A., Olmos, R., Fujita, H., & Herrera, F. (2020). Object detection binary classifiers methodology based on deep learning to identify small objects handled similarly: Application in video surveillance. *Knowledge-Based Systems*, Article 105590.

Prakash, A., Boochoon, S., Brophy, M., Acuna, D., Cameracci, E., State, G., Shapira, O., & Birchfield, S. (2019). Structured domain randomization: Bridging the reality gap by context-aware synthetic data. In *2019 International conference on robotics and automation* (pp. 7249–7255). IEEE.

Ravichandran, A., & Yegnanarayana, B. (1995). Studies on object recognition from degraded images using neural networks. *Neural Networks*, *8*(3), 481–488. http://dx.doi.org/10.1016/0893-6080(94)00077-Y, URL http://www.sciencedirect.com/science/article/pii/089360809400077Y.

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2015). You only look once: Unified, real-time object detection. arXiv:1506.02640.

Richter, S. R., Vineet, V., Roth, S., & Koltun, V. (2016). Playing for data: Ground truth from computer games. arXiv:1608.02192.

Romero, D., & Salamea, C. (2019). Convolutional models for the detection of firearms in surveillance videos. *Applied Sciences*, *9*(15), 2965.

Salazar, J. L., Soria, L. M., Álvarez-García, J. A., Enríquez, F., & Jiménez, A. R. (2019). Energy-efficient indoor localization wifi-fingerprint system: An experimental study. *IEEE Access*, *7*, 162664–162682.

Sommer, L. W., Schuchert, T., & Beyerer, J. (2017). Fast deep vehicle detection in aerial images. In *2017 IEEE winter conference on applications of computer vision* (pp. 311–319). http://dx.doi.org/10.1109/WACV.2017.41.

Taylor, G. R., Chosak, A. J., & Brewer, P. C. (2007). Ovvv: Using virtual worlds to design and evaluate surveillance systems. In *2007 IEEE conference on computer vision and pattern recognition* (pp. 1–8). IEEE.

Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., & Abbeel, P. (2017). Domain randomization for transferring deep neural networks from simulation to the real world. arXiv:1703.06907.

Tremblay, J., Prakash, A., Acuna, D., Brophy, M., Jampani, V., Anil, C., To, T., Cameracci, E., Boochoon, S., & Birchfield, S. (2018). Training deep networks with synthetic data: Bridging the reality gap by domain randomization. arXiv:1804.06516.

Unity (2020). URL https://unity.com/. (Accessed 20 Feb 2020).

Velastin, S. A., Boghossian, B. A., & Vicencio-Silva, M. A. (2006). A motion-based image processing system for detecting potentially dangerous situations in underground railway stations. *Transportation Research Part C: Emerging Technologies*, *14*(2), 96–113.

Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., & Van Gool, L. (2016). Temporal segment networks: Towards good practices for deep action recognition. In *European conference on computer vision* (pp. 20–36). Springer.

World's First Synthetic Gun Detection Dataset from Edgecase.ai (2020). URL https://blog.usejournal.com/worlds-first-synthetic-gun-detection-dataset-from-edgecase-ai-dbe3ea8eeb7e. (Accessed 20 Feb 2020).

Wu, Y., Kirillov, A., Massa, F., Lo, W.-Y., & Girshick, R. (2019). Detectron2. https://github.com/facebookresearch/detectron2.

Zhang, W., Wang, S., Thachan, S., Chen, J., & Qian, Y. (2018). Deconv R-cnn for small object detection on remote sensing images. In *IGARSS 2018 - 2018 IEEE international geoscience and remote sensing symposium* (pp. 2483–2486).