Computers, Materials & Continua
DOI:10.32604/cmc.2022.018785
Article

Tech Science Press

Weapons Detection for Security and Video Surveillance Using CNN and YOLO-V5s

Abdul Hanan Ashraf[1], Muhammad Imran[1], Abdulrahman M. Qahtani[2,*], Abdulmajeed Alsufyani[2], Omar Almutiry[3], Awais Mahmood[3], Muhammad Attique[4] and Mohamed Habib[5,6]

[1]Department of Computer Science, Shaheed Zulfikar Ali Bhutto Institute of Science and Technology, Islamabad, 44000, Pakistan
[2]Department of Computer Science, College of Computers and Information Technology, Taif University, Taif, 21944, Saudi Arabia
[3]College of Applied Computer Science, King Saud University (Almuzahmiyah Campus), Riyadh, 11543, Saudi Arabia
[4]Department of Computer Science, HITEC University, Taxila, 47080, Pakistan
[5]College of Computing and Informatics, Saudi Electronic University, Riyadh, Saudi Arabia
[6]Faculty of Engineering, Port Said University, Port Fuad City, Egypt
[*]Corresponding Author: Abdulrahman M. Qahtani. Email: amqahtani@tu.edu.sa
Received: 21 March 2021; Accepted: 10 May 2021

Abstract: In recent years, the number of Gun-related incidents has crossed over 250,000 per year and over 85% of the existing 1 billion firearms are in civilian hands, manual monitoring has not proven effective in detecting firearms. which is why an automated weapon detection system is needed. Various automated convolutional neural networks (CNN) weapon detection systems have been proposed in the past to generate good results. However, These techniques have high computation overhead and are slow to provide real-time detection which is essential for the weapon detection system. These models have a high rate of false negatives because they often fail to detect the guns due to the low quality and visibility issues of surveillance videos. This research work aims to minimize the rate of false negatives and false positives in weapon detection while keeping the speed of detection as a key parameter. The proposed framework is based on You Only Look Once (YOLO) and Area of Interest (AOI). Initially, the models take pre-processed frames where the background is removed by the use of the Gaussian blur algorithm. The proposed architecture will be assessed through various performance parameters such as False Negative, False Positive, precision, recall rate, and F1 score. The results of this research work make it clear that due to YOLO-v5s high recall rate and speed of detection are achieved. Speed reached 0.010 s per frame compared to the 0.17 s of the Faster R-CNN. It is promising to be used in the field of security and weapon detection.

Keywords: Video surveillance; weapon detection; you only look once; convolutional neural networks

1 Introduction

In the year 2016, the number of Gun-related incidents and deaths crosses over 250,000 across the globe [1], according to Ren et al. [2] there are over 1,013,000,000 firearms in the world and over 85% of them are in civilian hands. Forbes highlighted in the IBIS World's 2018 report, which stated that in the United States of America (USA) alone the sales of Guns were estimated around $28 Billion. Manual monitoring of security cameras is not enough to effectively detect and respond to such dangerous situations [3].

In the last few years, deep learning techniques and Convolutional Neural Networks (CNNs) have achieved great results in image detection, classification, segmentation and it's being used in several applications [4]. The advancements in technology and the latest innovative detection models such as YOLO, Faster R-CNN, VGG-16 have achieved satisfactory results [5]. The common challenges that are faced while weapon detection is the increase in complexity due to partial or full occlusion of gun [6] deformation and loss of information while transmission [7]. The rate of false-negative and false-positive also is an issue in weapon detection systems due to such sensitives systems being linked to alarms or such devices [8]. Weapon Detection systems need Real-time processing and fast response times due to their critical nature, so the research has to find and implement techniques that speed the processing time of weapon detection models [3,8].

The main problems such as the high number of false negatives and false positives occur due to challenges such as the similar shape and handling of non-weapon objects which are commonly handheld. Another major challenge is ensuring that the model doesn't fail to detect the weapon and has a very low false negatives rate. The model should also be able to avoid false positives from the background of these images and videos. The already presented models have a very high rate of false negatives when it comes to videos. Suppose if 10 people with weapons are interested to enter a building and out of these 10 only one person succeeded to enter in the building can lead to serious consequences. Therefore it is required to reduce the number of false negatives and false positives by improving techniques suggested by [8,9], while also expanding the range of weapons that can be detected to include rifles.

To achieve this goal, this study puts forth a model that can take advantage of the latest models such as YOLO which has very fast detection speeds. The main contributions of this work include reducing the number of false positives and negatives in the domain of Weapon Detection by using Gaussian blur to remove the background and only focusing on the area of Interest and its combined use with YOLOv5s with Stochastic gradient descent (SGD).

The remainder of this paper is organized as follows. Section 2 gives a brief analysis of the most related works to our work. Section 3 explains the methodology used in this research along with the proposed model. Section 4 described the experimental setup used in this work. Section 5 provides an analysis of the results and finally Section 6 conclusions.

2 Related Works

The related work is divided into two sections, section one focusing on weapon-related literature and the second section focusing on irregular object-shape detection which describes techniques that might become beneficial to our research. The purpose of the literature review is to discuss the existing software techniques/approaches proposed in contemporary studies in this domain. An allied purpose is to determine the research gaps and the key challenges linked to this area.

2.1 Weapon Detection and Security-Based Literature

Similar research conducted to be used for weapon detection includes the work done by [3] that used Faster-RCNN and VGG-16 on different datasets and videos from YouTube and test the sliding window approach, speed of detection reach 0.19 s per frame with the high precision rate but very low recall in terms of videos, while other research works [8] try to use preprocessing using dual cameras to blur the background using Global Block Matching algorithm, following the Area of interest methodology which increases their accuracy and lowers the false positive rate but adds more time to the detecting speed of the model. The research work presented by [10] focused on the effect of brightness with CNN based model, on the rate of detection in terms of weapons such as cold steal Knives and Blades, this was effective for indoor applications however different lighting conditions and reflections lower the accuracy in the outdoor application. Another methodology used by [9] was to train the model on similarly shaped objects and classes, to fine-tune the model to be able to stop confusing similarly shaped objects as weapons, this did result in better precision values but lower the recall due to the model confusing features of other objects to pistol features. Region proposal networks were used by [11], this approach trained multiple CNN models to detect single parts of the weapon such as the muzzle or the trigger and then took an average of the models to predict the existence of pistols in the images resulting in better accuracy but longer detection times. The research work done by [12,13] focused on detecting concealed weapons, this research work has to rely on the specialized images taken by passive mill meter wave (PMMW) cameras and using a CNN model to detect the grey objects as pistols which are similarly shaped. These systems provide effective results, however, the expensiveness of this technology is not applicable in households. The research done by [14] uses an ensemble of semantic neural networks to detect the firearms, basically delegating the different tasks to different neural networks and the average of the results depicting if a fire-arm was present or not. Faster-RCNN [15] model used to detect the social media image data for pistols and other weapons, using a two-pass convolutional network. An effort was made by [16] to improve weapon detection rate in single energy X-ray images by using pseudo coloring, using different color filters on the data to try and identify the weapons.

2.2 Irregular Shaped Object Detection and Supporting Literature

This section includes the use of different techniques and deep learning models for the detection of different objects, among which one research by [17] focuses on the smoke produced by Gunfire to detect the location of the fired Gun. Other research work done by [18] includes using the Faster-RCNN model to detect objects and pedestrians. Support vector machine (SVM) was used by [19] to do real-time clothing recognition from surveillance videos. A survey of Advances of Deep Learning using X-ray Security Imaging by [7] covered several CNN variations and other algorithms and compared their results in the domain of security and detection of a harmful object in luggage at airports. The research work presented by [5] focuses on face detection of low-quality images using the facial attributes presents in the image. Other research [6] focuses on multi-layer CNN features and exploiting the complementary strengths to use it for image retrieval. The author of [20] gives a detailed explanation of how we can visualize and understand the convolutional networks. Similarly [21] trying to detect harmful objects from special camera images using temperature differences in objects to find out the shape of the object with regards to the surroundings then used YOLOv3 to detect objects. This model performed well but the requirement of Passive Millimeter-Wave images makes the system not feasible for normal locations or homes that do not have such amount of funding or areas where the temperature is in the extreme colds or extremely hot conditions. The above literature review was used to try and identify

the best model for identifying objects in real-time speed with the best accuracy and pre-processing techniques used to enhance the results.

3 Proposed Framework

This research focuses on the speed of the algorithm as well as following the area of interest methodology supported by literature but using gaussian blur to remove the background, the YOLO-v5s algorithm is designed to have deal with the rate of false negatives and has faster speeds than the Faster R-CNN and other models used in all the research mentioned above.

As the weaknesses of earlier work have been discussed in the previous section so to overcome those weaknesses, the proposed framework is designed for accurate detection of Weapons. The model's main purpose is to reduce the number of false negatives and positives while giving a timely detection response. The proposed framework consists of the following steps (i) Input Dataset (ii) Data Preprocessing (iii) Model Architecture (CNN and YOLO) (iv) Performance Evaluation Metrics.

3.1 Data Preprocessing

Data pre-processing is an important phase of the data analysis activity which involves the construction of the final data set so it can be fed to deep learning algorithms. The pre-processing used in our model is to resize the images to 2 variations, 416×416 for the YOLO-V5 model because it only accepts variations of 32 and 240×240 for the CNN model. The second pre-processing technique that we use on the frames/images is to blur or remove the background from the images using different algorithms. Which in this model is Gaussian Blur operation, we opted to use this preprocessing rather than any other because of its speed compared to other techniques such as median filter which require sorting and slow down the operation. The Gaussian filter is a low-pass filter that removes the high-frequency components, the pixels nearest the center of the kernel are given more weight than those far away from the center. This averaging is done on a channel-by-channel basis, and the average channel values become the new value for the filtered pixel. The Gaussian blur is a type of image-blurring filter that uses a Gaussian function, which also expresses the normal distribution in statistics for calculating the transformation to apply to each pixel in the image. The formula of a Gaussian function is:

$$G(x, y) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{1}$$

In the above Eq. (1) where x is the distance from the origin in the horizontal axis, y is the distance from the origin in the vertical axis, and σ is the standard deviation of the Gaussian distribution.

3.2 Model Architecture (CNN and YOLO)

The below Fig. 1 shows the Architecture we used in our custom CNN model, the hyperparameters for the model include the Filter Size of 128 for the first convolution layer than 64 for the second and 32 for the third, and 16 for the last layer.

We also described the working of ReLU that we have used in our custom CNN after each Conv Layer, we have chosen to do Max Pooling with a filter size of 2×2, we have also described the working of the pooling layer, the stride for the Pooling and Conv Layers is set to default which is always 1.

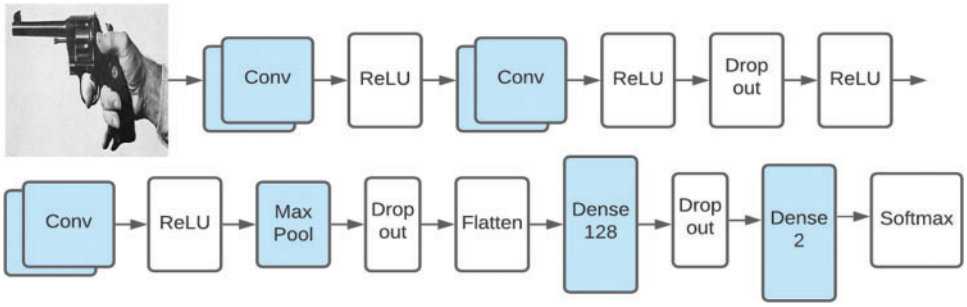

Figure 1: Custom CNN model

The drop out is set to 0.2, which in Keras means 20 percent, drop out is a strategy where, during preparation, randomly chosen neurons are overlooked and ignored, they are spontaneously dropped out. This result in their contribution to the activation of the neurons on the forward pass is momentarily excluded and any weight changes on the backward pass are not added to the neuron. This technique is used to prevent the model from overfitting. The Flatten operation then converts the multi-dimensions to a one-dimension vector and that is passed to the Dense 128 and Dense 2 fully connected neural network layer which output to the Softmax layer. The softmax layer is very handy because it transforms the scores to a distribution that is normalized and those probabilities can then be presented or used as an input to other systems.

The second model we used was the YOLOv5s model, its Architecture has three main parts like any other single-stage object detector, the Model Backbone, Model Neck, and the Model Head. Below Fig. 2 depicts the internal workings of the model and its structure.

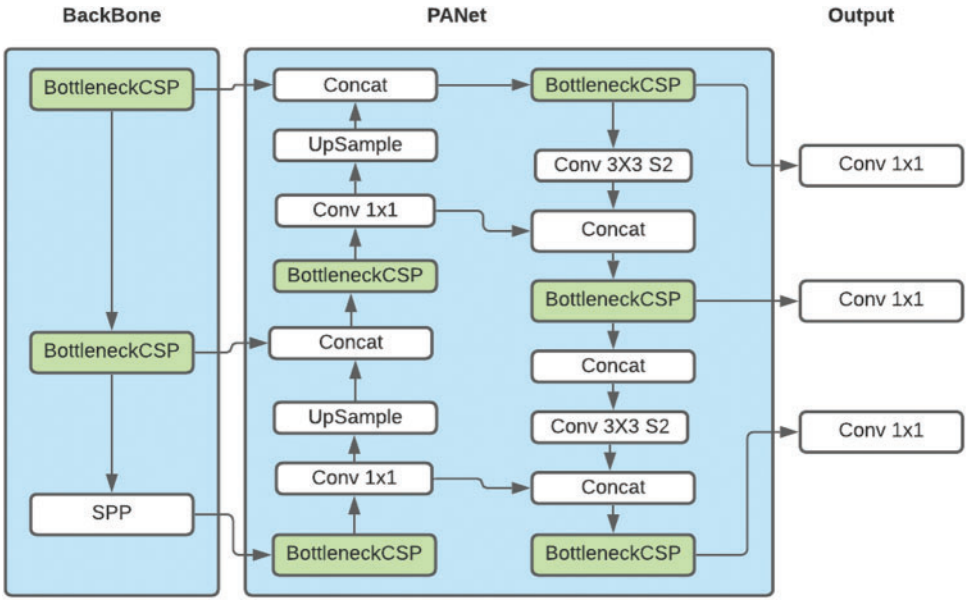

Figure 2: YOLOv5s model

The Model Backbone can be seen above in Fig. 2 is primarily used to extract the important features from the input image. Cross-stage Partial Networks (CSP) are used as the backbone

in YOLO v5 to extract the information-rich characteristics from an input image. With deeper networks, CSPNet has seen a substantial increase in processing time, the CSPNet is used for YOLO v5 because it removed the computational bottlenecks by evenly distributing the computation over all the convolutional layers to make sure that the arithmetic unit is not idle, the goal is to increase the utilization rate of each computation unit. Another advantage is the reduced memory costs, using the cross-channeling pooling to compress feature maps during the feature pyramid generation process, the object detector cuts down 75% memory usage.

The Model Neck is primarily used to create feature pyramids, the feature pyramids are useful in allowing the model to generalize well on object scaling. It helps to identify the same object with different sizes and scales. On unseen data, the Feature pyramids help the model to perform well. In our YOLO v5, Path Aggregation Network (PANet) is used for the neck to get feature pyramids.

The Model Head is mainly used to perform the final detection part. It applies anchor boxes on features and generates the final output vectors with class probabilities, bounding boxes, and the object scores.

The Activation function is most crucial in any deep neural network, YOLO v5 used in our model uses Leaky ReLU and Sigmoid [22]. The leaky ReLU activation function is used in the middle/hidden layers and the sigmoid activation function is used in the final detection layer. The Eq. (2) of Leaky ReLU, is given below where 'a' represents a small constant:

$$\mathrm{f(x)} = 1(\mathrm{x} < 0)(\alpha \mathrm{x}) + 1(\mathrm{x} >= 0)(\mathrm{x})\mathrm{f(x)} = 1(\mathrm{x} < 0)(\alpha x) + 1(x >= 0)(x) \tag{2}$$

The Optimization Function that our YOLO v5 uses for training is SGD because of the computation costs, as SGD uses small subsets and eliminates redundant and inefficient computations.

The Cost and Loss Function that our proposed model uses is the Binary Cross-Entropy with Logits Loss function which is provided by PyTorch for loss calculation of class probability and object scores. The equation of Binary Cross-Entropy is depicted in the below Eq. (3).

$$H_p(q) = -\frac{1}{N}\sum_{i=1}^{N} y_i * \log(p(y_i)) + (1 - y_i) * \log(1 - p(y_i)) \tag{3}$$

4 Experimental Setup

The experimental setup comprises different software and tools which are used to make the task of implementation easier along with the parameters, settings, and conditions used to train and test the model.

4.1 Machinery and Software Environment

For all the experiments conducted we have used a Mac Book pro Machine 2015 model with 16 GB of ram and 256 GB of SSD, the machine also includes 2.8 GHz Intel Core i7 processors with Intel Iris Pro 1536 MB for Graphics. Anaconda was used to manage all the tools, python version 3.7.3 was used along with Jupytor notebook to implement the CNN model and to code various methods. For YOLO v5 implementation google Colab was used which used python version 3.6.9 with 1.85 GB Graphics processing unit (GPU) provided by Google.

4.2 Selection of Datasets

The Dataset is a combination of the Open-sourced pistol dataset from the University of Granada; the dataset contains 3000 images of pistols along with their boundary box files in Pascal-VOC XML format which was used for the positive class of Pistols. This dataset has also been used in literature by [3]. The Dataset was increased by the addition of 12,887 Negative examples which are images that do not contain pistols. The resolution of images was 100×100 pixels and was sourced from [3] open-sourced from the University of Grandala. The dataset for the 3000 guns was mostly created from Internet Movie Firearms Database (IMFDB) dataset or Common Objects in Context (COCO) datasets. The dataset is divided into training, validation, and test set where 70% of the data is used for training and 20% used for the validation, and 10% is used for the test set. The total image count reaches 15,873, making it the largest existing dataset for pistol detection available as of date.

To test the model on videos, the data was sourced from YouTube, this decision was made to provide a fair comparison with existing research studies as they have used YouTube videos of old action films with weapon scenes from different angles to test their models. This study uses the same video samples, along with self-made videos. A 60-s video of myth-busters experimenting on guns which were taken from YouTube was used to check the speed and performance of the model on video data. We used Roboflow, which is an online website with collections of tools that help to organize, prepares, and improves your image and annotation training data. We use Roboflow to upload our dataset in VOC XML format and then Roboflow convert that data into text format which was required by our YOLO-V5s model. We also used Roboflow to pre-process our images into 416×416 which is recommended for the YOLO model.

4.3 Preprocessing Steps

The preprocessing steps for the implementation of YOLO-V5 and the custom CNN model are different. For the CNN model, the input image dimensions were resized to 240×240 using the image library from PIL, while for the YOLO-V5 the dataset was resized to 416×416 which is recommended for this model, Roboflow was used to stretched and resized these images, and then the dataset was downloaded. Gaussian blur was used to soften the image and to blur the background.

4.4 Platform and Frameworks

We opted Google Colab, which provided a 1.85 GB GPU to solve to achieve storage and good processing speed. Google Collaboratory was used in this research, which is a google based product that allows users to run code written in python on their browsers, it allows free access to GPU's, easy sharing of files, and notebooks via Google Drive and required very little to no configuration. Colab is used extensively by the machine learning community with applications such as working with TPU's, model training, or Tensorflow, etc.

Google drive is a limited free cloud storage option provided by googling it was used in this project to store the dataset for quick transfer to other google services such as google Colab. Google Drive can be mounted to google Colab notebooks and URLs can be used to transfer files from the drive to code. It was also very convenient for sharing folders among server people who work on the same project and synchronize files.

4.5 Model Architecture and Training

A problem we faced while trying to use Colab to train our model was that whenever we tried to upload the dataset of this size it would stop responding, after several tries, we decided to use Google Drive as cloud storage and refer to the link of the storage to the Colab notebook while having the Drive mounted. This resulted in very fast data transfer from our notebook and google drive. To train our YOLO-V5 model, the data was split into 3 subsets, test train, and validation. The training data consisted of 70% of the overall images while the validation set consisted of 20% and 10% of the test images were used later to evaluate the model. This split was chosen because it provides the best results as suggested by the literature review and the automated software of Roboflow. Since we used CNN and YOLO-V5 the architecture of the model differs and is provided below along with the hyperparameters used for the models. The Architecture of our CNN and YOLO-v5s along with their working are covered in Section 3.

4.6 Model Testing

The Testing data set consisted of 668 images with 329 images of weapons and the rest of the images consisting of non-weapon images. The trained model from the above step was loaded by using the Keras model library and the test directory in case of CNN was passed to the model after the images were resized to 240×240 on which the CNN was trained. The CNN model took an average of 0.050 s per image to process. For the YOLO-V5 the test images weren't resized yet and directly passed to the model, the model was very fast and resulted in 0.010 s on average per image. A YouTube video was introduced to the testing dataset and the YOLO-V5 due to its impressive speed was tested on a video of 60 s which consisted of 25 frames per second.

4.7 Model Assessment and Results Visualization

To evaluate both models, we decided to use the commonly used evaluation metrics used by most object detection and classification models from our literature review section such as accuracy, precision, recall, F1-score. The formulas for the calculation of these were given in Section 3. The models were compared based on their accuracy but a key factor was the speed of the model due to the application of this research in real-world threat scenarios which require fast responses to milliseconds per frame was a key factor for comparison. To Visualize the results of the models Tensor Board was used along with the utils library to plot the results in the form of graphs to get deeper knowledge through visualizations. TensorBoard is a suite of web applications for inspecting and understanding your TensorFlow runs and graphs. TensorBoard currently supports five visualizations: scalars, images, audio, histograms, and graphs. In this research, a tensor board was used to show how the number of the epoch's affected the results and at what point of the epoch's the model stopped learning and its curve stopped growing, it became pretty clear that 100–200 epochs for the YOLO-V5 and 60 epochs for the CNN model yielded the cap.

In this section we discussed the tools and hardware along with the different types of software and helper tools used to perform the experiments, the evaluation metrics used were also explained in detail in Section 3, the platform and frameworks have also been mentioned here with different software versions.

5 Results and Analysis

In this section, we explore the strengths and weaknesses of our model on the test images and YouTube videos. The below tables show the results of the models that were trained and validated on over 3,000 to 15,000 images and 4 base videos with an average of 24 frames per second.

5.1 Comparison with Existing Literature

In this section we will compare the results of our proposed model with results of models used by [3] and the pre-processing research by [8] as well as [9], we will compare based on the evaluation metrics mentioned in Section 3, the comparison includes both image and video data sets depicted in Tab. 1.

When we only train the YOLO-V5 model on 3000 images of pistols and then test on the 608 images test set the results of our model compared to the results of similar research using the same dataset shown in Tab. 1. From Tab. 1 it is revealed that when we follow the processes and dataset of [3] our YOLO-V5s model performs decent but the false positive rate is higher than the Faster R-CNN model implemented by [3] while our recall rate is impressive at 0.990. The proposed model has an impressive average speed of 0.010 s per frame/image which is 19 times faster compared to [3]. The YOLO-v5s model provides better speed of detection and quick response time which is needed in the case of weapon detection systems. we used a similar video of similar frames and evaluated our model against the best results of [3,8]. The results are clear that on images the Faster R-CNN used by [3] had performed slightly better, but had dropped the performance by almost 20 percent when it applies to videos data set, but our model performs even better on videos and has achieved 0.922 Precision compared to the 0.987 of [3] and scored 0.929 recall compared to the 0.374 of [3]. The last evaluation metric of time is the same for the images beating all other times and scoring 0.010 s per frame, while the Faster R-Convolutional Neural Network used in literature scoring around 0.17 to 0.19 s on the average frame.

Table 1: Results comparison with existing literature

Method	Dataset	No. of true positive	No. of false negative	No. of true negative	No. of false positive	Precision (%)	Recall (%)	F1 (%)	Time per frame
Olmos et al. [3]	Image dataset	304	0	247	57	84.2	100	91.4	0.19 s
Olmos et al. [3]	Video 2 of #627 frames	467	–	–	11	98.7	37.4	74.3	0.19 s
Olmos et al. [8]	Video 3 of #372 frames	331	44	99	15	95	88	91	NA
Pérez-Hernández et al. [9]	Video 4 #2188 frames	1113	–	–	158	87	44	61	NA
Proposed method	Image dataset	301	3	233	71	81	99	89.1	0.010 s
Proposed method	Video 2 of #627	514	43	31	39	92.2	92.9	92.5	0.010 s
Proposed method	Video 3 of #372 frames	308	23	18	23	93.3	94.4	93.8	0.010 s
Proposed method	Video 4 #2188 frames	1051	149	768	220	82.6	87.5	84.9	0.010 s

When we compare our results to the best results of [9] on video number 4 we see that the trend continues where the precision score is almost the same for our model and the Faster-RCNN

at 0.87 and our model scoring 0.825 but in case of true negative and recall the Faster-RCNN model always lags behind the proposed model, our model scored 0.875 which is leaps ahead of the competition.

When we compare the results of our model to another research done by [9] which used a Faster-RCNN model and utilized preprocessing techniques on the video frames to blur the background using disparity map and Global block matching algorithms, the result shows that the Faster-RCNN models have a lower rate of false-positive however, they have very high recall rate, which means that they are not very useful and often miss weapons in frames. Our proposed model has achieved a higher recall rate. The overall results also show that our model has achieved a 0.938 F1 score compared to the 0.91 of [8]. The time per frame has not been mentioned by that research probably because the pre-processing to blur the background of every frame takes a toll and further pushes the time of Faster R-CNN from 0.19 s to a very high time. We further tried to decrease the rate of false positives and false negatives by adding negative classes to the dataset while training the YOLO-V5 model and when these classes are added the results are depicted in the next section in Tab. 2.

Table 2: Results of YOLO-v5s trained on 15000 images

Reference	Dataset	Model	#TP	#FN	#TN	#FP	Precision (%)	Recall (%)	F1 (%)	Time per frame
Proposed model	Image dataset	YOLO-V5s	635	115	747	3	99.5	84.6	91.4	0.011 s
Proposed model	Video 2 #627 frames	VOLO-V5s	434	123	14	56	77.9	88.5	82.8	0.010 s
Olmos et al. [3]	Image dataset	Faster R-CNN	304	0	247	57	84.2	100	91.4	0.19 s
Olmos et al. [3]	Video 2 of #627 frames	Faster R-CNN	467	–	–	11	98.7	37.4	74.3	0.19 s

Fig. 3 visually depicts that the proposed model achieved a higher recall rate as compared to the existing research work. The closest in literature has achieved 88 percent with pre-processing being 88 percent. Recall in the case of weapon detection is more important for this research because missing weapons is more costly in terms of human causality. The overall F1 measure shows that the overall model performs better than the Faster-RCNN and preprocessing techniques used in literature before this.

Fig. 4 visually depicts the F1 score of the various research studies with our implemented YOLO-v5s model and provides an easy-to-see comparison. It can be seen that on video 2, our model performs better than the Faster-RCNN model and the same goes for video 4 by wide margins. The results of YOLO-v5 for video 3 are slightly better by 2% compared to the model used in the literature which used preprocessing to focus on the area of interest.

Fig. 5 depicts the speed comparison between the models used in literature for weapon detection to the speed of the Yolo-v5s model. The above chart shows that while the average speed per frame of Faster-RCNN is 0.17–0.19 s, the speed per frame of the YOLO-v5s model is 0.010 s, which is more than 10 times faster.

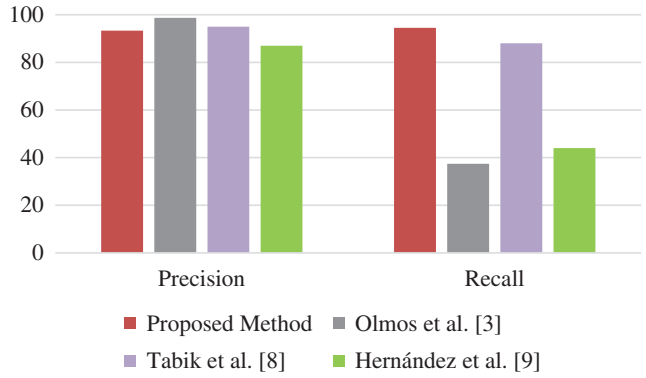

Figure 3: Evaluation of results using precision and recall

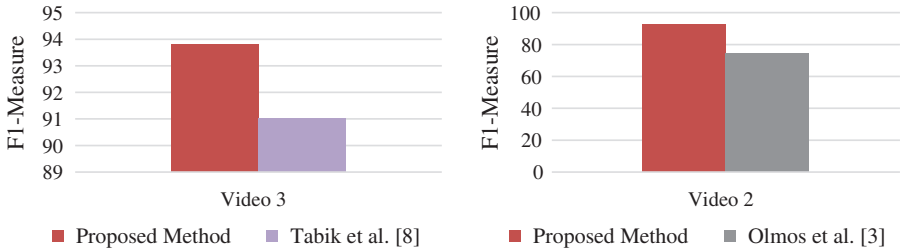

Figure 4: Comparison graphs of results using F1-measure

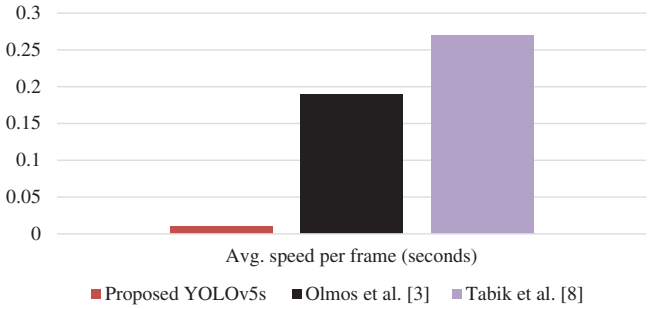

Figure 5: Speed comparison of faster R-CNN *vs.* Yolo-v5s

Tab. 2 show the results of the proposed model when we trained the YOLO-V5 model on over 15000 images including the 3000 images of pistols and other negative classes folder with random images of non-pistols. From row 1 of Tab. 1, it can be seen that there is an increase in the performance of the proposed model. Precision is better than the Faster R-CNN model used by [3], which was beating us when we only trained on the 3000 images. Overall F1-measure is slightly more than the Faster R-CNN by 0.001 points and has reached F1-measure of 91.44%, and yet the speed of the YOLO-V5s on average almost the same at 0.011 s per frame, 19 times faster than the Faster R-CNN.

In terms of videos, our proposed model's performance has increased, the false positive rate reaches 56 and the false negative has reached 123 frames, with an overall F1-score of 0.828. The model outperformed the Faster R-CNN in the F1 measure by 8 percentage points.

It seems that it is possible to improve the false-positive rates of YOLO-v5 by adding more training images that are often similar to pistols such as binoculars, which we found were being highly included in the images for the FP rate.

5.2 Comparison with Preprocessing on Video Frames

In this section, we will evaluate the results of our base model on the video data set without the use of preprocessing techniques on the frames. Tab. 3 shows the results of the proposed model after applying some preprocessing through Gaussian blur functions.

Tab. 3 shows that when we run the YOLO-v5s model on Video 1 of 1480 frames the TP rate is 535 but when we use the same model on the frames which had been pre-procced using Gaussian blur we see that the TP rate dropped by 9, while the False-negative rate has increased further leading to a Slight decrease in the recall. But at the same time, the False Positive rate has dropped quite significantly from 514 to 432 which have improved our precision by 4 percentage points. This leads to an overall improvement of 3 percent in the F1 measure from 0.656 to 0.686. In the real-life scenario, we cannot afford to miss any guns which makes recall critically important. If the trained model considered other objects as weapons, manual second checking of the object can confirm the presence of a weapon but we cannot miss a single weapon, which is the target of the proposed model, so depending on the scenario it would be better to use the preprocessing technique.

Table 3: Improvements from preprocessing

Test dataset	Preprocessing	Model	#TP	#FN	#TN	#FP	Precision (%)	Recall (%)	F1-measure (%)
Video 1 #1480 frames	N/A	VOLO-V5s	535	41	390	514	51	92	65.6
Video 1 #1480 frames	Gaussian blur	VOLO-V5s	526	50	472	432	55	91.3	**68.6**

Fig. 6 shows the result of the proposed model with preprocessing it can be seen that while the precision has increased the recall rate remains almost the same with a slight decrease, which leads to a higher F1-score after applying Gaussian blur on the video frames. Precision has increased, which leads to a higher F1-score. It is obvious from the results that the use of preprocessing has decreased the chances of mistakes and lowered the false positive rate. The reason behind the success and the speed of the proposed model are CSPNet because it removed the computational bottlenecks by making sure to increase the utilization rate of each computation unit. Another reason for reduced memory costs is the use of cross-channeling pooling to compress feature maps during the feature pyramid generation process. The object detector cuts down 75% memory usage along with the removal of background bring the rate of false detections further down which leads to better results.

In this section, after deciding YOLO-v5s performed better than our custom CNN model, we trained our model on 3000 images of pistols and then compared the results of our model with the

best results present in literature that uses the same dataset. We saw that our model was consistent in outperforming the Faster-RCNN models used in literature in terms of Recall, which boosted our F1-measure. The speed of the YOLO-v5s also outperformed all the models used so far for similar weapon detection problems. We trained our model on larger datasets and observe that the proposed model achieved higher precision and recall rate on video data set compared to the existing approaches. We used pre-processed video frames and use blurring techniques to improve our results, we observe that the recall rate remains the same but a big increase in precision value, which led us to an improved score of 0.68 F1 after preprocessing.

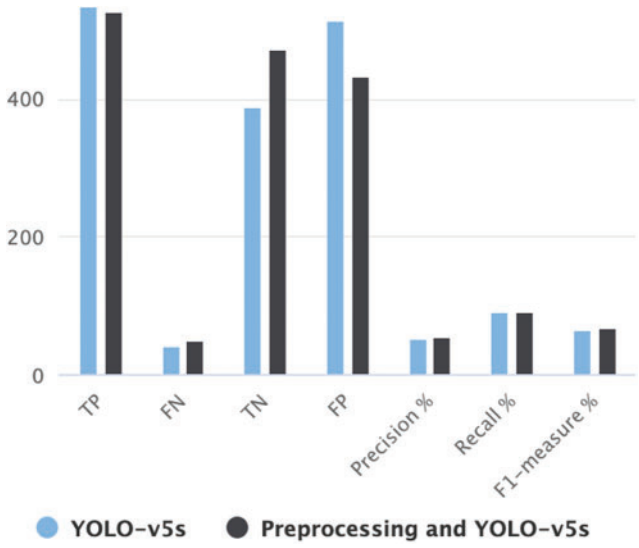

Figure 6: Result comparison after applying to preprocess

6 Conclusion and Future Work

In this work, we have proposed a model to detect pistols with the speed that can be used with alarm-based systems in applications of surveillance. We took the advantage of the latest models such as YOLOv5s that had very effective results and speed, we used it for the task of weapon detection that has not been done in any research article published so far. We further used the pre-processing technique of blurring the background with gaussian blur to improve our F1-score. The most promising results have been obtained from our YOLO-v5s model trained on the 3000-pistol image dataset provided by the University of Granada with the addition of 12,000 negative class images as well as tested on YouTube videos, our best results achieved 99% recall and 81% precision on images and 93% precision and 94% recall on the video, these results are better than the results achieved by similar research, especially the recall score of our model as well as the speed per frame with was 0.010 s which is 19 times faster than the Faster R-CNN model used by other research. This research can be used in combination with an alarm system to provide effective pistol detection. In the future, we will build on this model by using other preprocessing techniques such as brightness control, we also discovered several areas where the performance of the model can be increased by solving the issues, such as customized weapons which vary from the generic look of pistols as well as tried techniques that help distinguish between similar sized and shaped objects which are key challenges in the domain of weapon detection. We also hope

to try increasing their contrast and luminosity and also by enriching the training set with pistols in motion and with customized images and color on the pistols to try and reduce the number of false positives and false negatives.

Funding Statement: We deeply acknowledge Taif University for Supporting and funding this study through Taif University Researchers Supporting Project Number (TURSP-2020/115), Taif University, Taif, Saudi Arabia.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] N. Kurek, L. A. Darzi and J. Maa, "A Worldwide perspective provides insights into why a US surgeon general annual report on firearm injuries is needed in America," *Current Trauma Reports*, vol. 6, pp. 36–43, 2020.

[2] Y. Ren, C. Zhu and S. Xiao, "Small object detection in optical remote sensing images via modified faster R-CNN," *Applied Sciences*, vol. 8, no. 5, pp. 813–818, 2018.

[3] R. Olmos, S. Tabik and F. Herrera, "Automatic handgun detection alarm in videos using deep learning," *Neurocomputing*, vol. 275, no. 9, pp. 66–72, 2018.

[4] M. M. Ghazi, B. Yanikoglu and E. Aptoula, "Plant identification using deep neural networks via optimization of transfer learning parameters," *Neurocomputing*, vol. 235, no. 7, pp. 228–235, 2017.

[5] X. Shu, Y. Cai, L. Yang, L. Zhang and J. Tang, "Computational face reader based on facial attribute estimation," *Neurocomputing*, vol. 236, no. 10, pp. 153–163, 2017.

[6] W. Yu, K. Yang, H. Yao, X. Sun and P. Xu, "Exploiting the complementary strengths of multi-layer CNN features for image retrieval," *Neurocomputing*, vol. 237, no. 2, pp. 235–241, 2017.

[7] R. K. Tiwari and G. K. Verma, "A computer vision based framework for visual gun detection using harris interest point detector," *Procedia Computer Science*, vol. 54, pp. 703–712, 2015.

[8] R. Olmos, S. Tabik, A. Lamas, F. Pérez-Hernández and F. Herrera, "A binocular image fusion approach for minimizing false positives in handgun detection with deep learning," *Information Fusion*, vol. 49, no. 2, pp. 271–280, 2019.

[9] F. Pérez-Hernández, S. Tabik, A. Lamas, R. Olmos and F. Herrera, "Object detection binary classifiers methodology based on deep learning to identify small objects handled similarly: Application in video surveillance," *Knowledge-Based Systems*, vol. 194, pp. 212–223, 2020.

[10] A. Castillo, S. Tabik, F. Pérez, R. Olmos and F. Herrera, "Brightness guided preprocessing for automatic cold steel weapon detection in surveillance videos with deep learning," *Neurocomputing*, vol. 330, no. 9, pp. 151–161, 2019.

[11] S. Ren, K. He, R. Girshick and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in Neural Information Processing Systems*, vol. 39, pp. 91–99, 2015.

[12] B. Khajone and V. Shandilya, "Concealed weapon detection using image processing," *International Journal of Science and Engineering*, vol. 3, pp. 1–4, 2012.

[13] J. Xiong, S. Li, J. Yang, X. Xue and Z. Mao, "A novel Otsu method based on prior area information for concealed target detection in PMMW images," *MATEC Web of Conferences*, vol. 59, pp. 81–92, 2016.

[14] A. Egiazarov, V. Mavroeidis, F. M. Zennaro and K. Vishi, "Firearm detection and segmentation using an ensemble of semantic neural networks," in *European Intelligence and Security Informatics Conf.*, Norway, USA, pp. 70–77, 2020.

[15] J. Elsner, T. Fritz, L. Henke, O. Jarrousse and M. Uhlenbrock, "Automatic weapon detection in social media image data using a two-pass convolutional neural network," *European Law Enforcement Research Bulletin*, vol. 4, pp. 61–65, 2019.

[16] B. R. Abidi, Y. Zheng, A. V. Gribok and M. A. Abidi, "Improving weapon detection in single energy X-ray images through pseudocoloring," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 36, no. 6, pp. 784–796, 2006.

[17] Z. Zhou, I. C. Etinger, F. Metze, A. Hauptmann and A. Waibel, "Gun source and muzzle head detection," *Electronic Imaging*, vol. 2020, no. 8, pp. 187–181, 2020.

[18] G. L. Hung, M. S. B. Sahimi, H. Samma, T. A. Almohamad and B. Lahasan, "Faster R-CNN deep learning model for pedestrian detection from drone images," *SN Computer Science*, vol. 1, no. 2, pp. 1–9, 2020.

[19] M. Yang and K. Yu, "Real-time clothing recognition in surveillance videos," in *18th IEEE Int. Conf. on Image Processing*, NY, USA, pp. 2937–2940, 2011.

[20] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *European Conf. on Computer Vision*, NY, USA, pp. 818–833, 2014.

[21] L. Pang, H. Liu, Y. Chen and J. Miao, "Real-time concealed object detection from passive millimeter wave images based on the YOLOv3 algorithm," *Sensors*, vol. 20, no. 6, pp. 1678–1689, 2020.

[22] R. Li, J. Pan and L. Tang, "Single image dehazing via conditional generative adversarial network," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, Salt Lake City, USA, pp. 8202–8211, 2018.