International
Journal
Of Advanced Research in Engineering & Management (IJAREM)
ISSN: 2456-2033 || PP. 01-08

Analysis of Weapon Object Detection using Yolov4

Kyi Pyar Zaw
University of Computer Studies, Mandalay (UCSM)

Abstract: Weapon detection in surveillance settings is a critical task for ensuring public safety and security. This system presents an analysis of weapon detection using the state-of-the-art object detection model, You Only Look Once version 4(Yolov4). Yolov4 is a deep learning architecture known for its speed and accuracy in real-time object detection. The goal of this thesis is to investigate the effectiveness of Yolov4 in detecting weapons from images. The system involves training Yolov4 on a large dataset containing weapon categories and evaluating its performance on test images. The dataset used in this system comprises annotated surveillance images capturing various real-world scenarios. The training is carried out on a high-performance computing infrastructure, utilizing GPU acceleration for efficient model convergence. The evaluation phase involves analyzing the performance metrics of the trained model on the test images. The experimental results demonstrate the effectiveness of Yolov4 in accurately detecting weapons from image dataset. The demonstrated performance of Yolov4 highlights its potential for enhancing security measures in various domains, including airports, public spaces, and critical infrastructure facilities. The system was tested and was found to achieve high accuracy in detecting weapons with low false positive rates.

Keywords: Weapon Detection, Deep Learning, Annotated Surveillance Images, Object Detection,

1. Introduction

Now a days, weapons are being used around the world. In society, protection and security have become a major concern. Weapon detection plays a crucial role in ensuring public safety and security, especially in surveillance scenarios. The advancements in deep learning techniques have paved the way for more accurate and efficient object detection models, such as Yolov4 (You Only Look Once version 4). Yolov4 is renowned for its ability to perform real-time object detection with high precision and recall rates. This study aims to analyze the effectiveness of Yolov4 in detecting weapons from images. By leveraging the power of deep neural networks, the goal is to develop a robust weapon detection system that can accurately identify potential threats in real-world surveillance scenarios. The Yolov4 model, with its exceptional speed and accuracy, is well-suited to address these challenges and provide efficient weapon detection capabilities. The system involves training Yolov4 on the custom datasets specifically created for weapon detection. The datasets include annotated surveillance images, where weapons are labeled with bounding boxes to provide ground truth for training and evaluation. By training the model on such the datasets, it learns to detect weapons in different contexts. To optimize the performance of Yolov4, a comprehensive training process is conducted. This process includes hyperparameter tuning, adjusting model architecture parameters, and utilizing data augmentation techniques to enhance the model's ability to generalize to unseen weapon instances. The training process is carried out on GPU acceleration for efficient model convergence. After training, the performance of the Yolov4 model is evaluated on a set of test images that were not used during training. The evaluation focuses on key performance metrics, such as precision, recall, and average precision, to assess the model's accuracy in weapon detection. This system contributes to the advancement of weapon detection technologies and lays the foundation for implementing robust security systems in various domains.

2. Literature Review

Object detection is a computer vision task aimed at identifying and localizing multiple objects within an image or video frame [1]. It involves two primary aspects: accurately classifying object categories and precisely defining their spatial extents through bounding boxes. The task typically relies on deep learning models like Yolo (You Only Look Once), which employ convolutional neural networks to extract hierarchical features from input data. These features are then used to predict object classes and bounding box coordinates simultaneously.

Object detection models learn to recognize patterns and features at different scales, enabling them to detect objects of varying sizes while leveraging techniques like anchor boxes to enhance localization accuracy [2]. This theoretical framework enables automated and efficient object recognition, finding applications across various fields, including surveillance, autonomous driving, and image analysis [3], [4].

Weapon detection is a crucial field of study with significant applications in security and safety domains. Yolo employs an Artificial Neural Network (ANN) methodology for object detection in images. The network partitions the image into multiple regions and makes predictions for each region, providing bounding box coordinates and corresponding probabilities [5]. These predicted bounding boxes are subsequently compared with their associated probabilities.

Among the existing works on weapon detection, many of them primarily focus on employing complex CNN networks for object detection of guns. However, these approaches often overlook the practicality of real-world deployment [6]. Some algorithms claim to be real-time, but their reliance on expensive GPU or CPU machines hinders their scalability for large-scale use.

The research introduced in gun detection, where the Yolov3 algorithm is utilized and conducted a comparative analysis of false positives and false negatives against the Faster RCNN algorithm [7]. To enhance the performance, the researchers curated a comprehensive dataset of handguns, encompassing all possible angles, which was then merged with the ImageNet dataset. The merged dataset was subsequently employed to train the Yolov3 algorithm, leading to improved detection results.

The researchers conducted a comprehensive comparative analysis between two state-of-the-art models, namely Yolov3 and Yolov4, for weapons detection [8]. To facilitate the training process, they curated a specialized weapons dataset by collecting images from Google Images and incorporating various assets. The images were manually annotated in different formats to meet the requirements of Yolo and other models, such as text format and XML format, respectively. Both versions of the models were trained on this extensive weapons dataset, and their results were subsequently tested to perform a thorough comparative analysis.

The author S. Khan et al. incorporates the Yolov5 deep learning architecture and a specialized dataset of pistol images [9]. This innovative system aims to detect pistols in real-time video streams and promptly send email alerts to the administrator upon a positive detection. Deep learning has revolutionized the field of object detection, enabling the development of highly accurate and efficient algorithms to identify and localize objects in images and videos [10]. Object detection is a fundamental task in computer vision with numerous practical applications, such as surveillance, autonomous vehicles, robotics, and medical imaging. Traditionally, object detection relied on handcrafted features and machine learning classifiers. However, deep learning, particularly Convolutional Neural Networks (CNNs), has emerged as the dominant approach due to its ability to automatically learn hierarchical representations from raw data [11]. You Only Look Once (Yolo) is another popular real-time object detection approach that divides the input image into a grid and predicts bounding boxes and class probabilities for each grid cell. It provides faster detection but may sacrifice some accuracy compared to other methods [12]. Faster R-CNN introduced the concept of region proposal networks (RPNs) to efficiently generate candidate regions for objects [13]. This two-stage approach achieves high accuracy with improved speed compared to its predecessors. RetinaNet addressed the issue of class imbalance in object detection by using a focal loss function. This approach helps focus on hard-to-detect objects, leading to more balanced and accurate predictions [14].

In Yolov3, the backbone forms the foundational feature extraction architecture, utilizing a Darknet-53 network to extract hierarchical features from input images [15]. The neck, PANet, plays a pivotal role by fusing and enhancing feature representations from various scales, contributing to Yolov3's improved detection accuracy across a wide range of object sizes and categories [16].

3. Overview of the System

The design of the system is shown in Fig 1. Before feeding an image into a deep neural network for object detection, the following step is to preprocess the image. This preprocessing step ensures that the input data is in a suitable format for the network's computations. One preprocessing technique involves resizing the image to a fixed size, (416x416), which standardizes the input dimensions for the model. Additionally, images are normalized by scaling their pixel values to 0 and 1. This normalization aids in stabilizing the training process and testing process and improves the model's ability to learn meaningful patterns from the data.

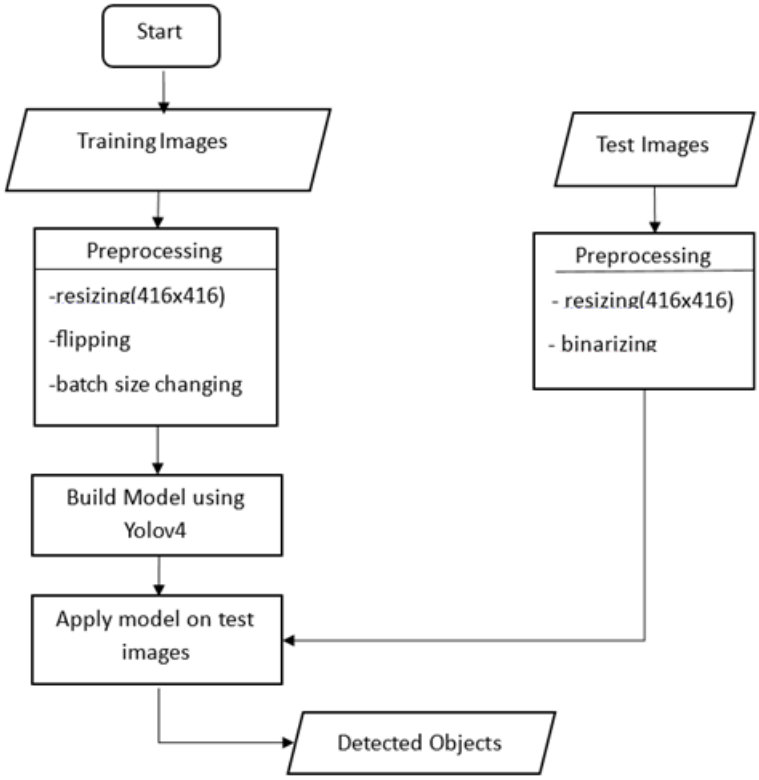

Figure 1: Overview of the system

3.1 Preprocessing

Preprocessing in Yolo (You Only Look Once) refers to the series of steps taken to prepare input data (images) before feeding them into the Yolo model for object detection. These steps ensure that the input data is in a suitable format and scales for the model's processing. Typical preprocessing steps in Yolo include Resizing, Normalization, Data Augmentation, Batching, Label Encoding and Data Type Conversion: Finally, the preprocessed data is loaded into the Yolo model for training.

3.2 Object Detection

Object detection in computer vision involves identifying and localizing objects of interest within an image. Convolutional Neural Networks (CNNs) are widely used in object detection tasks. CNNs excel at processing grid-like data, such as images, and can effectively learn and extract features for detecting objects. In the context of object detection, CNN-based architectures, such as Yolo (You Only Look Once) and Faster R-CNN (Region-based Convolutional Neural Network), are commonly employed.

3.2.1 Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN), is a class of deep neural networks that is widely used for computer vision tasks, including image classification, object detection, and image segmentation. CNNs are particularly effective in processing grid-like data, such as images, due to their ability to capture spatial relationships and local patterns. The main building blocks of a CNN are convolutional layers, which are responsible for learning and extracting relevant features from the input data. These layers consist of filters (also called kernels) that convolve with the input to produce feature maps. Convolutional layers typically employ operations such as convolution, activation functions, and pooling to process the input data.

3.2.2 Yolov4 (You Only Look Once version 4)

Yolov4 (You Only Look Once version 4) is an advanced object detection algorithm that builds upon the success of previous Yolo versions and incorporates several improvements to achieve state-of-the-art

performance in terms of accuracy and speed. Yolov4 offers different architectural variants, including Yolov4-CSP, Yolov4x, and Yolov4-tiny, with varying trade-offs between speed and accuracy.

In Yolov4 and other deep learning models, "weights" refer to the learned parameters of the model or the model's knowledge during the training process. These weights represent the numerical values that adjust the behavior of the model's layers and neurons, allowing it to make predictions on new, unseen data. During training, the model is optimized to learn these weights by minimizing a defined loss function, typically through techniques like gradient descent. The weights capture the patterns, features, and relationships in the training data that are relevant for making accurate predictions.

The batch size determines how many samples are processed in each iteration before updating the model's weights. Larger batch sizes can lead to faster convergence, but they may also require more memory resources. Max batch refers to the maximum batch size that can be used during the training process. It represents the largest number of training examples that can be processed together in a single iteration of training.

In this system, the value of the max_batch hyperparameter is derived by multiplying 2000 with the number of classes within the dataset. Since the Dataset used in this system has 4 classes, the max_batch value becomes 8000. At every 1000 iterations of the training process, the model's weights are saved to a designated file. These weight files are loaded into the model to initialize its parameters, allowing it to make predictions on new images.

3.2.3 Complete Intersection Over Union (CIoU)

The CIoU metric provides several advantages over traditional IoU. Firstly, it penalizes inaccurate predictions with higher precision, as it considers the bounding box regression distance. This property encourages the model to make more accurate and precise predictions. Secondly, CIoU has a bounded range [-1, 1], making it easier to interpret the quality of the predictions. A value close to 1 indicates a highly accurate prediction, while a value close to -1 indicates a poor prediction. Lastly, CIoU has shown to be more robust to small object detections, making it suitable for various computer vision tasks, including object detection and instance segmentation.

• Calculate the Intersection over Union (IoU) as usual:

$$\text{IoU} = (\text{Area of Intersection}) / (\text{Area of Union}) \quad (1)$$

• Next, compute the distance between the centers of the predicted and ground truth bounding boxes:

$$d_x = \text{center\_x(predicted)} - \text{center\_x(ground truth)} \quad (2)$$

$$d_y = \text{center\_y(predicted)} - \text{center\_y(ground truth)} \quad (3)$$

• Finally, the Complete Intersection over Union (CIoU) is computed as follows:

$$C = \left(w(predicted) - W(ground_{truth})\right)^2 + \left(h(predicted) - H(ground_{truth})\right)^2 \quad (4)$$

• Calculate the smallest enclosing box that encloses both the predicted and ground truth bounding boxes:

$$\text{CIoU} = \text{IoU} - \frac{(d_x^2 + d_y^2)}{C} \quad (5)$$

3.2.4 Non-maximum suppression (NMS)

Non-maximum suppression (NMS) is a technique commonly used in object detection algorithms, including Yolov4, to remove redundant or overlapping bounding box predictions and retain only the most confident and accurate detections. When an object detection model generates multiple bounding box predictions for the same object instance, these predictions may partially overlap or cover the same region. To avoid redundant detections, NMS is applied to select the most relevant bounding box predictions based on their confidence scores and the degree of overlap.

The steps in NMS are as follows:

1) Sort the bounding box predictions based on their confidence scores in descending order.
2) Initialize an empty list to store the selected bounding boxes.
3) Iterate over the sorted list of bounding box predictions:
    a) Select the bounding box with the highest confidence score and add it to the selected list.
    b) Calculate the Intersection over Union (IoU) between this selected bounding box and the remaining bounding boxes in the list.

c) Remove the bounding boxes that have an IoU greater than a predefined threshold value. These overlapping bounding boxes are considered redundant.

4) Repeat steps 3 until all bounding boxes have been processed.
5) Return the final list of selected bounding boxes, which represents the non-redundant and most confident detections.

NMS helps to filter out duplicate detections and retain only the most relevant ones, improving the precision and reducing false positives in object detection tasks. By eliminating redundant predictions, NMS helps to generate a cleaner output with a reduced number of bounding boxes. The specific implementation of NMS may vary depending on the object detection framework or library being used. However, the general concept described above apply to most NMS implementations, including those used in Yolov4.

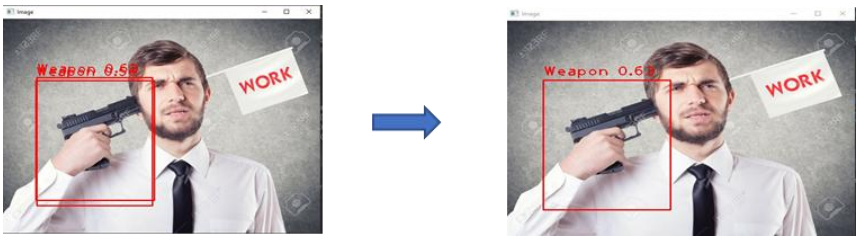

Figure 2. Before Non-maximum suppression and After Non-maximum suppression

4. Dataset Description

The training dataset and the testing datasets are separated into 80 : 20 ratio of the dataset. The weapon images are collected from the open image dataset v6 and Kaggle.com. Some images are collected from publicly available websites of some of the paper. The data collection contains the different pixel images with the .jpg, .png and .jpeg file extension. There are 6822 total images to detect the weapons in dataset. The dataset includes four classes with weapons such as pistol, rifle and knife and not weapon such as dollar. The system involves a hyperparameters experiment. There are two variables that used in the system: variable of standard, which serves as the system's benchmark as shown in Table 1, and variable input, which serves as an experiment's hyperparameter for comparing the performance of the Yolo architecture as shown in Table 2.

Table 1: Standard variables

Class	Filters	Network Size	Max Batches	Steps	Batch
4	27	416x416	8000	6400,7200	64

Table 2: Variable Input Experiment

Subdivision	Data Augmentation
16	Flipped

5. Experimental Results

In this dataset, 6822 images are used and total bounding boxes are 7807. There are four classes (1369 knife objects, 3613 pistol objects, 1948 rifle objects and 877 not_weapon objects) in dataset. This dataset is experimented on weight 6000, 7000, 8000, 9000 and 10000 using various pairs of confidence threshold value, non-maximum suppression threshold value and CIoU threshold value. The different pairs is defined as follow:

Table 3: Defining Pairs with Different Threshold Values

Name	confidence threshold	non-maximum suppression threshold	CIoU threshold
Pair_1	0.25	0.4	0.0
Pair_2	0.25	0.4	0.1
Pair_3	0.5	0.4	0.0
Pair_4	0.5	0.4	0.1

In this system, the results are evaluated on mean Average Precision (mAP), Precision, Recall, F1score and Accuracy.

$$mAP = \sum_{q=1}^{Q} \frac{AveP(q)}{Q} \quad (6)$$

Where *Q* is the number of queries in the set and AveP(q) is the average precision (AP) for a given query, q.

$$\text{Precision} = \frac{TruePositive}{TruePositive + FalsePositive} \quad (7)$$

$$\text{Recall} = \frac{TruePositive}{TruePositive + FalseNegative} \quad (8)$$

$$\text{F1score} = 2 * \frac{(precision * recall)}{precision + recall} \quad (9)$$

$$Accuracy = \frac{True\ Positives + True\ Negatives}{True\ Positives + False\ Positives + True\ Negatives + False\ Negatives} \quad (10)$$

Table 4: Experimental Results using Pair_1

Weight	TP	TN	FP	FN	mAP	Accuracy	F1 score	Recall	Precision
6000	1314	140	140	107	0.64	0.85	0.91	0.92	0.90
7000	1318	139	131	104	0.65	0.86	0.92	0.93	0.91
8000	1318	142	129	101	0.64	0.86	0.92	0.93	0.91
9000	1318	143	128	100	0.64	0.87	0.92	0.93	0.91
10000	1318	144	123	99	0.65	0.87	0.92	0.93	0.91

According to Table 4, the accuracy using weight 8000, 9000 and 10000 are the same but the accuracy using weight 8000 is the best accuracy because the number of true positives are greater than the other weights and that of false negatives are less than the other weights.

Table 5: Experimental Results using Pair_2

Weight	TP	TN	FP	FN	mAP	Accuracy	F1 score	Recall	Precision
6000	1283	140	171	138	0.61	0.82	0.89	0.90	0.88
7000	1287	139	162	135	0.62	0.83	0.90	0.90	0.89
8000	**1290**	**142**	**157**	**129**	**0.62**	**0.83**	**0.90**	**0.91**	**0.89**
9000	1287	143	159	131	0.62	0.83	0.90	0.91	0.89
10000	1285	143	157	133	0.62	0.83	0.90	0.91	0.89

According to Table 5, the accuracy using weight 7000, 8000, 9000 and 10000 are the same but the accuracy using weight 8000 is the best accuracy because the number of true positives are greater than the other weights and that of false negatives are less than the other weights.

Table 6: Experimental Results using Pair_3

Weight	TP	TN	FP	FN	mAP	Accuracy	F1 score	Recall	Precision
6000	1260	137	89	164	0.63	0.85	0.91	0.88	0.93
7000	1285	136	83	140	0.64	0.86	0.92	0.90	0.94
8000	1285	140	80	136	0.65	0.87	0.92	0.90	0.94
9000	1283	140	87	138	0.64	0.86	0.92	0.90	0.94
10000	1284	140	90	137	0.65	0.86	0.92	0.90	0.93

According to Table 6, the accuracy using weight 7000, 9000 and 10000 are the same and the accuracy using weight 8000 is the best accuracy because the number of true positives are greater than the other weights and that of false negatives are less than the other weights.

Table 7: Experimental Results using Pair_4

Weight	TP	TN	FP	FN	mAP	Accuracy	F1 score	Recall	Precision
6000	1232	137	117	192	0.61	0.82	0.89	0.87	0.91
7000	1254	136	114	171	0.62	0.83	0.90	0.88	0.92
8000	1259	140	106	162	0.62	0.84	0.90	0.89	0.92
9000	1256	140	114	165	0.62	0.83	0.90	0.88	0.92
10000	1256	140	118	165	0.62	0.83	0.90	0.88	0.91

According to Table 7, the accuracy using weight 7000, 9000 and 10000 are the same and the accuracy using weight 8000 is the best accuracy because the number of true positives are greater than the other weights and that of false negatives are less than the other weights.

6. Conclusion

This system is used for the detection of weapons from custom dataset. The experimental results of the system depend on the various confidence thresholds, nms thresholds and ciou thresholds. In this system, various confidence threshold (0.25.0.5), nms threshold (0.4) and ciou threshold (0, 0.1) are used. Among them, Pair_1: confidence threshold 0.25, nms threshold 0.4 and ciou threshold 0 achieves the best results 0.91 predcision, 0.93 recall, 0.92 F1 Score and 0.87 accuracy values were achieved on the above thresholds. The results obtained from the evaluation of the weapon detection system using Yolov4 demonstrated its potential for real-time application in surveillance environments. The high detection accuracy and real-time performance of the model make it a valuable tool for enhancing public safety and security. Fig 3 shows the detected images that are resulted by using the above threshold values.

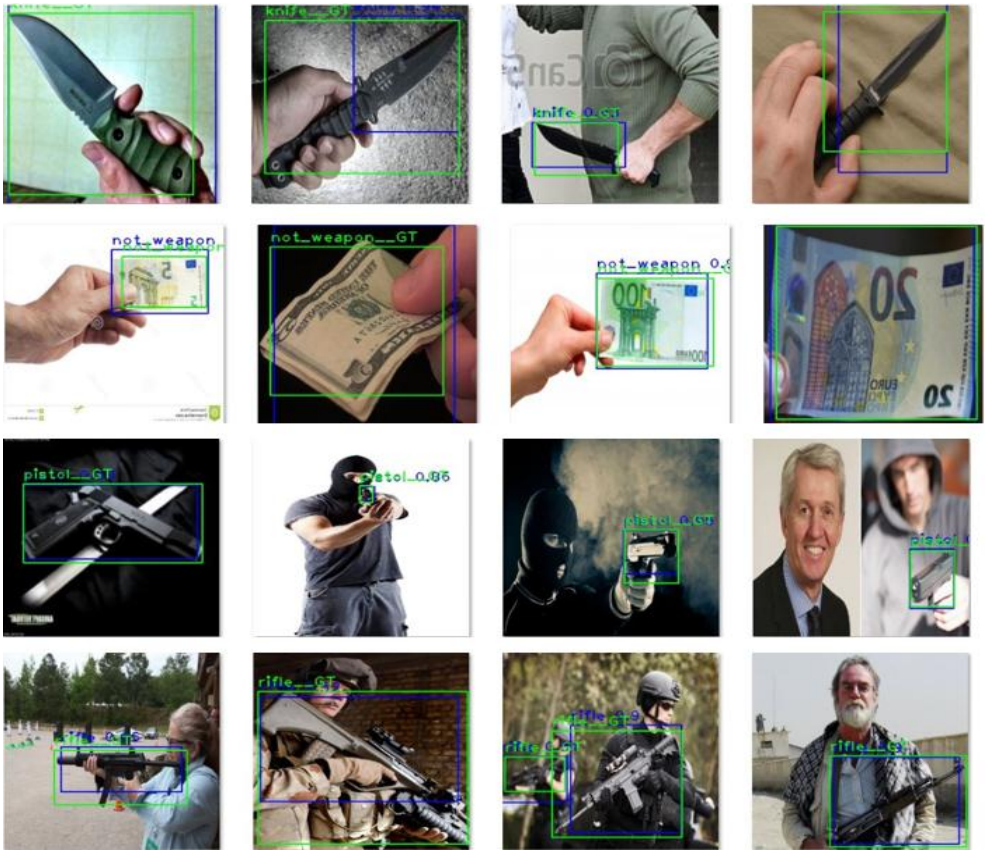

Figure 3: Sample detected images using Pair_1

References

[1] A. Hanan Ashraf et al., "Weapons Detection for Security and Video Surveillance Using CNN and Yolo-V5s," Comput. Mater. Contin., vol. 70, no. 2, pp. 2761–2775, 2022, doi: 10.32604/cmc.2022.018785.

[2] M. T. Bhatti, M. G. Khan, M. Aslam, and M. J. Fiaz, "Weapon Detection in Real-Time CCTV Videos Using Deep Learning," IEEE Access, vol. 9, pp. 34366–34382, 2021, doi: 10.1109/ACCESS.2021.3059170.

[3] A. P. Saputra, "Waste Object Detection and Classification using Deep Learning Algorithm: Yolov4 and Yolov4-tiny," 2021.

[4] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal Speed and Accuracy of Object Detection." arXiv, Apr. 22, 2020. Accessed: Jul. 14, 2023. [Online]. Available: http://arxiv.org/abs/2004.10934

[5] N. Thakur, P. Nagrath, R. Jain, D. Saini, N. Sharma, and J. Hemanth, "Object Detection in Deep Surveillance," In Review, preprint, Nov. 2021. doi: 10.21203/rs.3.rs-901583/v1.

[6] A. P. Jana, A. Biswas, and Mohana, "Yolo based Detection and Classification of Objects in video records," in 2018 3rd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), Bangalore, India: IEEE, May 2018, pp. 2448–2452. doi: 10.1109/RTEICT42901.2018.9012375.

[7] A. Warsi, M. Abdullah, M. N. Husen, M. Yahya, S. Khan, and N. Jawaid, "Gun Detection System Using Yolov3," in 2019 IEEE International Conference on Smart Instrumentation, Measurement and Application (ICSIMA), Kuala Lumpur, Malaysia: IEEE, Aug. 2019, pp. 1–4. doi: 10.1109/ICSIMA47653.2019.9057329.

[8] T. S. S. Hashmi, N. U. Haq, M. M. Fraz, and M. Shahzad, "Application of Deep Learning for Weapons Detection in Surveillance Videos," in 2021 International Conference on Digital Futures and Transformative Technologies (ICoDT2), Islamabad, Pakistan: IEEE, May 2021, pp. 1–6. doi: 10.1109/ICoDT252288.2021.9441523.

[9] S. Khan, M. Sayyed, S. Yadav, B. Bhalerao, and A. D. Patil, "Weapon Detection and Alarm System Using Yolov5," vol. 7, no. 3.

[10] C. V. Amrutha, C. Jyotsna, and J. Amudha, "Deep Learning Approach for Suspicious Activity Detection from Surveillance Video," in 2020 2nd International Conference on Innovative Mechanisms for Industry Applications (ICIMIA), Bangalore, India: IEEE, Mar. 2020, pp. 335–339. doi: 10.1109/ICIMIA48430.2020.9074920.

[11] F. Harrou, M. M. Hittawe, Y. Sun, and O. Beya, "Malicious attacks detection in crowded areas using deep learning-based approach," IEEE Instrum. Meas. Mag., vol. 23, no. 5, pp. 57–62, Aug. 2020, doi: 10.1109/MIM.2020.9153576.

[12] C. Kumar B., R. Punitha, and Mohana, "Yolov3 and Yolov4: Multiple Object Detection for Surveillance Applications," in 2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT), Tirunelveli, India: IEEE, Aug. 2020, pp. 1316–1321. doi: 10.1109/ICSSIT48917.2020.9214094.

[13] M. Abraham, N. Suryawanshi, N. Joseph, and D. Hadsul, "Future Predicting Intelligent Camera Security System," in 2021 International Conference on Innovative Trends in Information Technology (ICITIIT), Kottayam, India: IEEE, Feb. 2021, pp. 1–6. doi: 10.1109/ICITIIT51526.2021.9399597.

[14] N. Bordoloi, A. K. Talukdar, and K. K. Sarma, "Suspicious Activity Detection from Videos using Yolov3," in 2020 IEEE 17th India Council International Conference (INDICON), New Delhi, India: IEEE, Dec. 2020, pp. 1–5. doi: 10.1109/INDICON49873.2020.9342230.

[15] W. A. Ezat, M. M. Dessouky, and N. A. Ismail, "Evaluation of Deep Learning Yolov3 Algorithm for Object Detection and Classification," Menoufia J. Electron. Eng. Res., vol. 30, no. 1, pp. 52–57, Jan. 2021, doi: 10.21608/mjeer.2021.146237.

[16] U. Nepal and H. Eslamiat, "Comparing Yolov3, Yolov4 and Yolov5 for Autonomous Landing Spot Detection in Faulty UAVs," Sensors, vol. 22, no. 2, p. 464, Jan. 2022, doi: 10.3390/s22020464.