Recent Trends in Intensive Computing
M. Rajesh et al. (Eds.)

doi:10.3233/APC210191

Parallel Deep Learning Framework for Video Surveillance System

Mayuri KARVANDE [a,1] , Apoorv KATKAR [a] , Nikhil KOLI, [a] , Amit JOSHI [a] and Suraj SAWANT [a]

[a] *Dept. of Computer Engineering and IT, College of Engineering Pune (COEP), India*

Abstract. In today's world, the security of every individual has become an important aspect. There is a need for constant monitoring in public places. A Manual operating camera system is an unreliable and very basic and poor method for this purpose. Intelligent Video Surveillance is an approach where multiple CCTVs conscantly record the scenes and proper algorithms are deployed in order to detect and monitor activities. Deep Learning frameworks and algorithms like Kera's, YOLO, Convolutional Neural Networks or backbones for image detection like VGG16, Mobile net, Resnet101 have been used for human and weapon detection. The paper focuses on deep learning techniques and threading to collectively develop a Parallel Deep Learning Framework for Video Surveillance that aims at striking the right balance between accuracy and system performance or stability. Threading is used in terms of implementation of a uniquely proposed Dynamic Selection Algorithm that uses two backbones for object detection and switches between them based on the queue status for achieving system stability. A uniquely designed logistic regression filter is also implemented that boosts the system performance.

Keywords. Surveillance, deep learning, regression filter, ResNet

1. Introduction

In recent times, machine learning and deep learning algorithms have created a lot of buzz. These days surveillance systems play a vital role in public security. CCTV cameras are used in almost all sectors where security has importance. However, most of these systems are very straightforward and have a simple capacity of recording with limited capabilities. Today, when an incident related to public security happens, pre-recorded clips are analyzed by humans. As this is a post-incident analysis, it delays the response to the incident which is the most valuable time in which the crime could be solved the fastest. Surveillance systems are not used efficiently with human interventions and automation is limited. Considering the rapidly increasing availability of cameras and the advance- ments in computing environment, it is worth taking a look at a method to detect events automatically without compromising on system stability as video accounts for huge data volumes [1]. This paper aims at proposing a solution for the above problem.

[1] Mayuri KARVANDE ,Dept. of Computer Engineering and IT, College of Engineering Pune (COEP), Pune-411005, Maharashtra, India; E-mail: karvandema17.comp@coep.ac.in.

The paper proposes a new surveillance model using existing deep learning architectures and parallelism techniques. It aims at using the full power of the recent advancements in technology so as to decrease human intervention through the learning capabilities of the deep learning model [2]. It would help in throwing alerts which would not have been possible as it would be practically impossible for a single human to cover the wide vision of the camera and analyse it continuously throughout the day. For example, 24-hour surveillance video input at 20 fps will account for more than 15 lakh frames [3]. It is practically impossible for a human to be attentive for these large time durations, thereby making a human-based approach for the analysis of live surveillance videos in efficient and impractical. It is tedious and time-consuming. Naturally implementing such a complex system in real-time will put considerable stress on the system resources in order to work as required. When implementing such a solution, the system should be reliable. The system reliability cannot be ignored and its solution is discussed in the form of a Dynamic Selection Algorithm in the paper. It is based on the use of threading and a queue. The unique logistic regression filter analyses if there is a chance for a human to exist in the frame and then passes it to the model to analyse only if required [4]. This architecture helps us to only analyse frames that have a significant probability of a human existing. We call this architecture Filter Before using YOLO (FBYOLO). Now, the queue is fed sampled frames from the video feed. Here based on its current state, the queue switches between two deep learning models which are running on two different threads. The two powerful networks of YOLOv3 - ResNet and MobileNet have been used in this method- ology [5]. The queue architecture is explained in subsequent chapters. All the proposed objectives are implemented using one or more deep learning models and one will be selected among them based on the time required for each model to predict so as to arrive at the most relevant and practical solution [6]. We aim at a solution that can be used at various sites like malls, offices, homes, etc for systems with limited computing powers like that of a normal gaming laptop.

2. Literature Review

There has been significant work in the domain of video surveillance systems. Accuracy and time are two equally important aspects for designing any system. Different deep learning algorithms are considered and the algorithm which handles both the aspects and yields better results is selected at a time. This dynamic security-level control algorithm works towards maximizing accuracy related to recognition performance and at the same time minimizing operation time related to the system stability [6]. Surveillance is essential to analyze suspicious activities, most of which occur in crowded public places. Object recognition, crowd analysis and violence detection become the important parameters in this case. Deep learning techniques are capable of incorporating these things. Big data needs to be dealt with in this situation. Human detection and posture identification are incorporated using YOLO, VGG-16, Convolutional Neural Network (CNN). Deep CNN gives better results in terms of crowd analysis. These deep learning models can be combined and implemented on a powerful Graphics Processing Unit (GPU) to get a better Surveillance system [1]. YOLOv2 has proved to be an accurate and faster method than the Region Based CNN (R-CNN) for object detection with a

greater number of fps in the field of image processing. Around 78.6 map and 155 fps can be reached with this method [7]. Another version, YOLOv3 using K-means clustering over the frames of the dataset, called the tiny-YOLOv3, is an improved method for real-time object detection [8]. YOLOv3 is also a better technique over Faster R-CNN in terms of weapon detection. Handguns with various shapes are accurately detected in various scenes considering differment scales, rotations and angles [9]. Weapons and other theft-related small objects can also be detected using a two-level CNN binary classifier. The candidate regions selected in the first level undergo binary classification in the second level based on One-Versus-All or One-Versus-One [10].

Apart from weapon detection, human detection is also important for surveillance. Extraction of information from a foreground-background 2D grey image using Adaptive Gaussian Mixture Model (AGMM) along with the ConvNets YOLO models precisely determines the presence of a human [11]. Background subtraction, optical flow and extraction of spatiotemporal parameters using the Gaussian model helps in the detection of any human in motion [12]. In order to reduce the time required for computationally heavy real-time surveillance tasks, a lightweight CNN technique has been proposed which affordably detects humans on the edge device. This technique has been inspired by the Single Shot Multi-Box Detector (SSD) method [13]. The surveillance system should incorporate the detection of multiple people. W4 is an approach that operates on grey images and creates models of people on the basis of shape analysis, body parts and tracking people. It identifies activities taking place between objects and people using Gaussian models [14]. To achieve parallelism in surveillance, a multi-layer edged Distributed Intelligent Video Surveillance (DIVS) system has been designed using deep learning. Task-level and model-level parallel training methods are used to balance the workload [15].

3. Proposed Solution

The proposed solution is a way to provide various kinds of surveillance with maximized accuracy and performance. CCTV cameras capture and store the instantaneous footages [2]. The camera needs to be positioned such that it covers a wide area and every relevant aspect. The instantaneous video footages are supplied to the system and processed further. This work predominantly focuses on the two major objectives. The first objective is the Stealth Mode Movement Detection which is the most basic level of the surveillance system. The presence of humans in the video feed are identified and analyzed further. While the second important objective is Weapon Detection. Weapons like guns, knives and masks are often used by thieves and robbers to cover their faces and secretly ac- accomplish their goals. Most of the crimes involve gun violence [9]. Thus, the second objective recognizes weapons, particularly guns, from the video feed. These objectives are achieved using deep learning with parallel programming. The proposed methodology, named FBYOLO, is incorporated to minimize the time required to process a frame and maximize the accuracy of the prediction done by the model.

This methodology is based on the 'Dynamic Selection Algorithm' that selects one deep learning framework for optimizing performance-accuracy tradeoff [6]. Figure 2 depicts the block diagram for the system. It is as follows: the input CCTV video

footage is supplied to the model for processing. This video is then sampled into frames in order to process further. These frames are then fed to a simple logistic regression network. This regression is used as a filter. The regression filter is supposed to eliminate the true negative frames or the frames which don't have a human or a target object in it [4]. The FBYOLO recognizes the frames containing human and target weapons and separates them. The logistic regression network smartly calculates the probability of each frame based on the presence or absence of the required objects or human beings. The loss function is used in order to calculate the probabilities. The logistic loss function continuously tries to reduce the cost giving more optimistic results [16]. The frames which accurately predict the absence of the target are filtered out. The filtration saves the processing done for true negatives and thus enhances the frame processing rate of the model. This additional feature helps to get the relevant frames and increase both the accuracy and performance of the system.

After this initial step, the filtered frames are fed to the YOLOv3 networks. Different deep learning YOLOv3 networks having different time-accuracy characteristics work on these frames [17]. The backbones of these YOLOv3 networks used in the proposed method are ResNet and Mobilenet. The YOLOv3 algorithm is an efficient and faster method for real-time object detection [8]. The YOLOv3 algorithm creates bounding boxes for the image and calculates their probabilities. Based on these probabilities, the image classes are identified [18]. The Dynamic Selection Algorithm has been designed to select the appropriate YOLOv3 backbone model. This algorithm solves the time-accuracy trade-off depending on the number of frames [6]. Figure 1 depicts the pseudocode of this proposed algorithm. All the incoming filtered input frames are fed to a queue. A thread is created in order to enqueue the frames. The total capacity of the queue is represented as *qmax*. The threshold capacity of the queue is around 60%. The queue is considered to be full when its capacity reaches 90%. A separate thread is created to dequeue the frames one-by-one for processing over the YOLOv3 networks. This thread runs parallelly with the enqueue thread. In this dynamic algorithm, the ResNet backbone is set initially. While the frames are fed to the queue, the backlog is constantly compared to the threshold value. If the current queue backlog is more than the threshold value, the backbone network is switched to MobileNet. MobileNet is faster than Resnet but slightly less accurate [19]. MobileNet is used because there are a greater number of incoming frames and hence faster processing is required [20]. Whereas, if the current queue backlog is less than the threshold value, the backbone network is switched to ResNet. In this case, slower processing is permissible due to a smaller number of incoming frames. Higher detection accuracy is achieved using ResNet [21]. ResNet has a greater number of layers in its network and hence is a deeper network than MobileNet which makes it more accurate [22]. However, if the queue is almost full, then all the frames are discarded except for one. The analysis and processing is started again with the MobileNet backbone. Thus, the appropriate network is used in the current condition of queue backlog considering the computation time and accuracy parameters. This concludes the object detection for the given sampled frame and the explanation of the block diagram. The post-processing is done on the frame to draw a box around the detected object (viz. human, weapon).

Algorithm 1 Pseudocode for Dynamic Selection Algorithm

```
q = queue()
qmax = queue.capacity()
threshold = 0.6 * qmax
qalmostfull = 0.9 * qmax
backbone = ResNet
while frames are feeded to the q do
    if q.current() <threshold then
        backbone ← ResNet {more accuracy but slower}
    else if q.current() >= threshold then
        backbone ← MobileNet {less accuracy but faster}
    else {q.current() >= qalmostfull}
        discard all frames except one frame f#
        backbone ← MobileNet
        q ← f#
    end if
end while=0
```

Figure 1. Algorithm 1 – Pseudocode for Dynamic Selection Algorithm

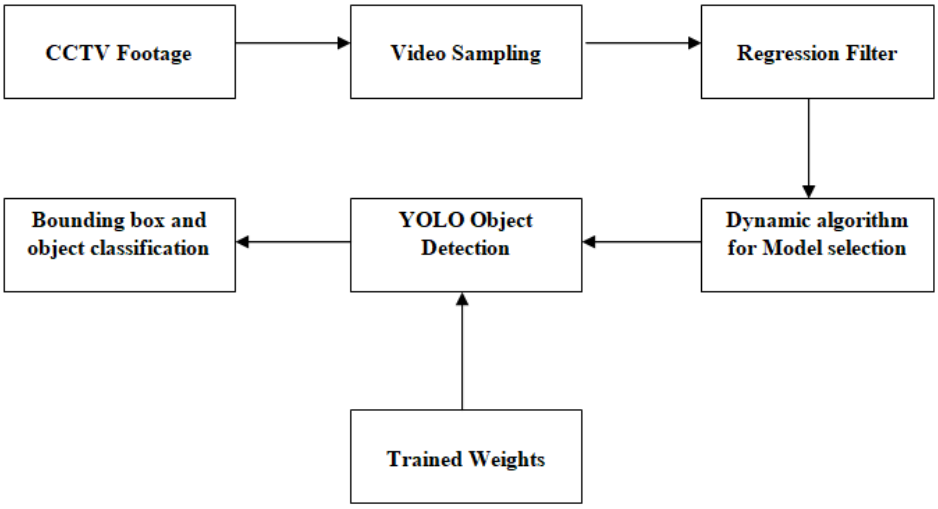

Figure 2. Block diagram of the proposed system

4. Experimental Setup

All the predictions are done on a gaming laptop system with configuration an Intel Core i5-7300 HQ CPU @ 2.5GHZ, 8GB DDR4 RAM @ 2400 MHz, NVIDIA GTX 1050 GPU. The operating system used is Windows 10. For rapid execution of tasks, hardware accelerators are required. A GPU that supports the CUDA framework is

required in order to achieve parallelism. The most important component of the system is the feed from a video capturing source. High-quality cameras should be positioned to get footage from proper angles. The input feed is scaled down to 512 x 512 pixels for further processing, to have a minimum output resolution of 512 x 512 pixels.

The programming environment required for the system is Python3. It is an easily accessible language with inbuilt libraries. Initially, threading was used for parallelism but it was observed that Keras is not compatible with threading. Hence, the multiprocessing library in python is used for parallelising the prediction using multiple processes as needed and explained in the Dynamic Selection Algorithm. The Queue module from the multiprocessing library is also used. Keras, a library that supports Deep Learning in Python, is used in the implementation. Tensorflow is a library that contains models of Machine Learning and Deep Learning. OpenCV is a fundamental library in real-time computer vision. It is used for video capture and analysis. It is used for human detection, object detection and image processing.

The dataset used for human detection for the first objective is TUD Brussels [23]. The size of the dataset is around 3GB and around 1476 images are used for the first objective. Out of these 1327 images were used for training. The remaining 149 images were used for testing. The dataset used for weapon detection is obtained from Soft Computing and Intelligent Information Systems which is a research group from the University of Granada [24]. It contains around 3000 images of which 1300 are used for training and 1000 for testing purpose.

5. Results and Discussion

The experiments have been carried out on the Nvidia GTX 1050 system. For human detection, images from TUD Brussels dataset have been trained. It has been observed that the frame per second rate for ResNet was 5.2fps. Whereas the rate for MobileNet is found to be 13.5fps. Thus, ResNet is slower than MobileNet. However, ResNet (88%) is found to be more powerful than MobileNet (81%) in terms of accuracy. In order to solve this issue and to maintain equilibrium between the speed and accuracy, an input video of about 8 seconds with a frame rate of 40 fps was fed to the queue at the rate of 6 frames per second. The total number of frames are around 331 out of which none of the frames is filtered out by FBYOLO since all frames are confirmed to be true positive. Metrics of the regression filter can be calculated from its confusion matrix. Left half of the Figure 4 indicates the confusion matrix. The precision, recall and accuracy calculated for the filter are 0.89, 1 and 91 % respectively. Out of 331 frames, the Dynamic Selection Algorithm chose 264 frames to be processed by ResNet, while the remaining 67 by MobileNet. The overall time for prediction is around 55.76 seconds which gives a frame rate of 6fps. This rate can be calculated using the following equation-

$$Overall\ Frame\ Rate = \frac{Total\ Frames}{\frac{ResNet\ frames}{ResNet\ \ frame\ rate} + \frac{MobileNet\ frames}{MobileNet\ frame\ rate}} \quad (1)$$

Similarly, for an overall rate of 7fps, the prediction requires 51.70 seconds. In this case, the algorithm selected ResNet for 203 frames and MobileNet for 128 frames. Thus, the Dynamic Selection Algorithm helps in running a slower but more accurate model - ResNet at a higher fps with reduced overhead by using a lighter and faster model like MobileNet to predict the backlogged frames. This ultimately results in faster and accurate prediction. For weapon detection, an input video of 6 seconds of 30fps was fed to the queue at the rate of 7 frames per second. Dynamic Selection Algorithm chose Resnet (6 fps) to predict 124 frames out of total 186 frames. For the rest 62 frames MobileNet was chosen (15 fps). Figure 3 indicates the output with FPS values for both the detectors. Similar metrics for different FPS values for Human Detector are tabulated in Table 1. For all the above results the queue size was set to 10 and the threshold value was set to 40% of $qmax = 4$. The accuracy, precision and recall score for the human and weapon detector on specific FPS values using Resnet and MobileNet are tabulated in Table 2. For weapon detector, better results are achieved on the dataset as compared to the research group from the University of Granada. From the above discussion it is quite obvious that the more accurate ResNet will take longer time to process the frames as compared to MobileNet. Hence at lower FPS values the Dynamic Selection Algorithm will choose ResNet for processing majority of the frames. As FPS value increases, ResNet will not be able to keep up with it and the Dynamic Selection Algorithm will choose the faster MobileNet to process majority of the frames. These results are tabulated in graphical form in the right half of the Figure 4.

Table 1. Human Detector Model Summary

FPS	ResNet101 (frames processed)	MobileNetV2 (frames processed)	Total Time Taken in seconds
5	331	0	65
6	264	67	55.76
7	203	128	51.7
8	160	171	43
9	114	217	41
10	94	237	38.70
11	30	301	30.72
12	8	323	29.86

Table 2. Metrics of Human Detector on TUD Brussels Dataset (Input Size = 512*512) and Weapon Detector on Guns Dataset (Input Size = 416*416)

Detector	Model	Accuracy	Precision	Recall	Frame Rate
Human Detector	ResNet101	88.50%	0.9887	0.8940	5.3
	MobileNetV2	81.52%	0.9883	0.82299	13
Weapon Detector	ResNet101	83.86%	1	0.8386	6
	MobileNetV2	81.91%	0.9989	0.8198	15

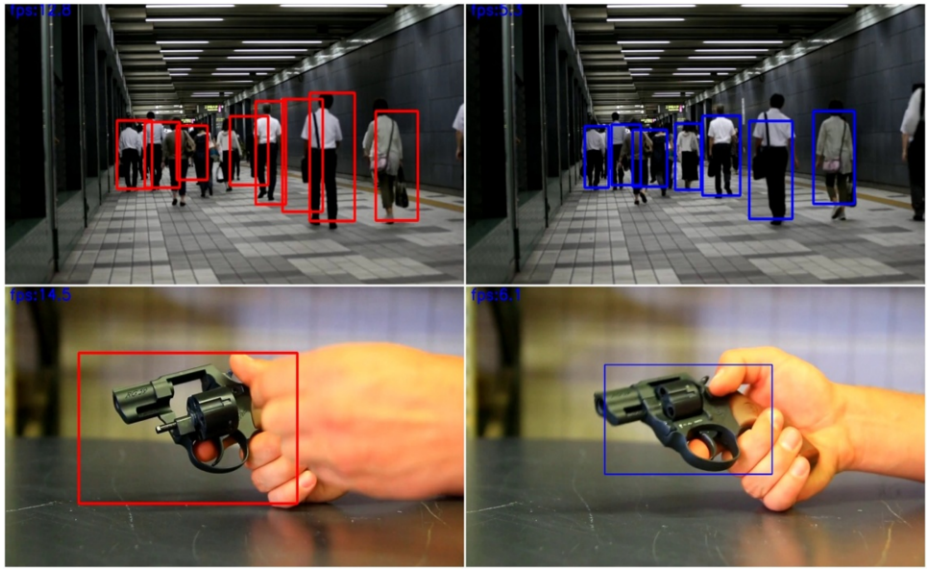

Figure 3. MobileNet vs ResNet with current FPS

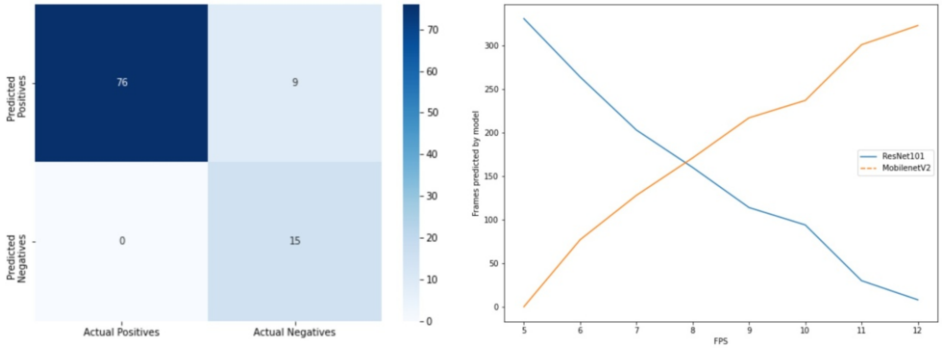

Figure 4. Left: Confusion Matrix of Regression Filter, Right: Model Selection by Dynamic Selection Algorithm

6. Conclusion and Future Scope

Initially, threading was used for introducing parallelism but it was observed that Kera's is not compatible with threading. Hence, multiprocessing was used for parallelizing the prediction using multiple processes as needed and explained in the Dynamic Selection Algorithm. It is clearly observed from the results that with increase in the fps value the number of frames predicted by MobileNet increases which hampers the accuracy of the system. It is balanced by selection using the Dynamic Selection Algorithm at suitable fps value that balances the accuracy and performance so as to achieve an efficient and stable output from the system.

Mobilenet has a small overhead as compared to ResNet. Its weights are of size 60 MB whereas ResNet has a higher overhead with its weights being of size 500MB. The accuracy of ResNet is significantly higher as compared to Mobilenet. Even on a normal system with GTX 1050, values like 7fps are achieved using ResNet with the Dynamic Selection Algorithm.

While selecting a model for a particular application one tends to use a model that has higher accuracy. But when the speed at which the model does the prediction also has an impact on the application it needs to be taken into account. Both high accuracy and faster predictions are not possible at the same time. By using the proposed Dynamic Selection Algorithm, a trade-off between speed and accuracy can be achieved by changing the threshold values and the size of the queue as required for the application. The speed of models with high accuracy can be boosted by using the Dynamic Selection Algorithm.

References

[1] Sreenu, G., Durai, M. S. (2019). Intelligent video surveillance: a review through deep learning techniques for crowd analysis. Journal of Big Data, 6(1), 1-27.

[2] Kardas, K., Cicekli, N. K. (2017). SVAS: surveillance video analysis system. Expert Systems with Applications, 89, 343-361.

[3] AlGhamdi, M. A., Khan, M. A., AlMotiri, S. H. (2015, October). Automatic motion tracking of a human in a surveillance video. In 2015 IEEE First International Smart Cities Conference (ISC2) (pp. 1-4). IEEE.

[4] Feng, J., Xu, H., Mannor, S., Yan, S. (2014). Robust logistic regression and classification. Advances in neural information processing systems, 27, 253-261.

[5] Yadav, N., Binay, U. (2017). Comparative study of object detection algorithms. International Research Journal of Engineering and Technology (IRJET), 4(11), 586-591.

[6] Kim, J., Mo, Y. J., Lee, W., Nyang, D. (2017, August). Dynamic security-level maximization for stabilized parallel deep learning architectures in surveillance applications. In 2017 IEEE Symposium on Privacy-Aware Computing (PAC) (pp. 192-193). IEEE.

[7] Du, J. (2018, April). Understanding of object detection based on CNN family and YOLO. In Journal of Physics: Conference Series (Vol. 1004, No. 1, p. 012029). IOP Publishing.

[8] Yi, Z., Yongliang, S., Jun, Z. (2019). An improved tiny-yolov3 pedestrian detection algorithm. Optik, 183, 17-23.

[9] Warsi, A., Abdullah, M., Husen, M. N., Yahya, M., Khan, S., Jawaid, N. (2019, August). Gun detection system using YOLOv3. In 2019 IEEE International Conference on Smart Instrumentation, Measurement and Application (ICSIMA) (pp. 1-4). IEEE.

[10] Pe´rez-Herna´ndez, 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, 194, 105590.

[11] Van Tuan, N., Nguyen, T. B., Chung, S. T. (2016, October). ConvNets and AGMM based real-time human detection under fisheye camera for embedded surveillance. In 2016 International Conference on Information and Communication Technology Convergence (ICTC) (pp. 840-845). IEEE.

[12] Paul, M., Haque, S. M., Chakraborty, S. (2013). Human detection in surveillance videos and its applications-a review. EURASIP Journal on Advances in Signal Processing, 2013(1), 1-16.

[13] Nikouei, S. Y., Chen, Y., Song, S., Xu, R., Choi, B. Y., Faughnan, T. R. (2018, July). Real-time human detection as an edge service enabled by a lightweight cnn. In 2018 IEEE International Conference on Edge Computing (EDGE) (pp. 125-129). IEEE.

[14] Haritaoglu, I., Harwood, D., Davis, L. (2000). W4: Real-time surveillance of people and their activities. 22 (8): 809–830.

[15] Chen, J., Li, K., Deng, Q., Li, K., Philip, S. Y. (2019). Distributed deep learning model for intelligent video surveillance systems with edge computing. IEEE Transactions on Industrial Informatics.

[16] Vovk, V. (2015). The fundamental nature of the log loss function. In Fields of Logic and Computation II (pp. 307-318). Springer, Cham.
[17] Mao, Q. C., Sun, H. M., Liu, Y. B., Jia, R. S. (2019). Mini-YOLOv3: real-time object detector for embedded applications. IEEE Access, 7, 133529-133538.
[18] Redmon, J., Farhadi, A. (2018). Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767.
[19] Bianco, S., Cadene, R., Celona, L., Napoletano, P. (2018). Benchmark analysis of representative deep neural network architectures. IEEE Access, 6, 64270-64277.
[20] Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., ... Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861.
[21] He, K., Zhang, X., Ren, S., Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).
[22] Vaddi, S., Kumar, C., Jannesari, A. (2019). Efficient object detection model for real-time UAV applications. arXiv preprint arXiv:1906.00786.
[23] Wojek, C., Walk, S., Schiele, B. (2009, June). Multi-cue onboard pedestrian detection. In 2009 IEEE Conference on Computer Vision and Pattern Recognition (pp. 794-801). IEEE.
[24] Olmos, R., Tabik, S., Herrera, F. (2018). Automatic handgun detection alarm in videos using deep learning. Neurocomputing, 275, 66-72.