AN EXPERT WEAPON IDENTIFICATION IN SECURITY SYSTEMS WITH CONVOLUTION NEURAL NETWORK (CNN)-BASED SSD AND FASTER RCNN ALGORITHM

[1*]Mr.J. Sudhakar, Dr. N.Satheesh[2], Dr. Govinda Rajulu[3]
Dr. B. Rajalingam[4]

[2,3,4]Associate Professor, Dept Of CSE ,
St.Martin's Engineering College, Dulapally, Kompally, Secunderabad, Telangana.

[1]Assistant Professor,Dept.of CSE
St.Martin's Engineering College, Dulapally, Kompally, Secunderabad, Telangana

Abstract-

Because of an increase in crime rates during crowded events or maybe secluded places, security is always a top priority in any profession. Computer vision may be used to solve a wide variety of difficulties, including the identification and monitoring of irregularities. Because of the growing demand for safety, security, and personal property protection, video surveillance systems that can recognise and understand scenes and anomalous occurrences are becoming increasingly important in intelligence monitoring. In this article, a convolution neural network (CNN)-based SSD and faster RCNN algorithms are employed to achieve automated gun (or) weapon identification. The suggested approach makes use of two datasets. One dataset contains pre-classified photos, whereas the other contains images that have been manually categorised. However, practical usage of the results is contingent on a trade-off between speed and accuracy, which both approaches accomplish.

Keywords--Firearm identification, computer vision, quicker RCNN, SSD, CCTV, and Artificial Intelligence (AI).

I. INTRODUCTION

Weapon detection, also called anamoly detection, is the identification of irregular, unexpected, unpredictable, or unusual occurrences or things that are not judged to be a regularly occurring event or a regular item in a pattern or items included in a dataset, and consequently diverge from present patterns. A pattern that departs from a set of typical patterns is referred to as an anomaly. As a consequence, anomalies are impacted by the phenomena of interest [3] [4].

Object detection distinguishes instances of different types of objects using feature extraction and learning approaches or models [6]. Detecting and categorising firearms properly is the goal of the implementation under consideration. Precision is also a worry for me, since an erroneous alert might have serious effects [11] [12]. Choosing the appropriate strategy involves a careful balance between accuracy and rapidity. Figure 1 displays the deep learning-based weapons identification method. Frames are culled from the video stream being fed into the system. A frame-difference approach is used in order to construct a bounding box and begin the process of object detection.

The flow of object identification and tracking may be shown in Figure 2. In order to train and feed the object detection algorithm, a dataset must first be created, trained, and fed into the system. For gun detection, a suitable detection technique (SSD or fast RCNN) was selected depending on the application. RCNN and Single Shot Detection (SSD) are two machine learning models that may be used to tackle a detection issue [2] [9] [15].

II. Implementation

A. Resources or components used for implementation.

- OpenCV 3.4: Open Source Computer Vision Library Version 3.4.
- Python 3.5 is a high-level programming language that is used in a variety of image-processing applications.
- COCO Dataset: A dataset made up of commonobjects and their labels.
- Tensorflow 1.1 and Anaconda
- NVIDIA GeForce 820M GPU-GeForce is a brand of graphics processing units designed by Nvidia.

1. **Dataset Specifications**

Specifications for video

- System Configuration—Intel i5 7th Generation (4Cores)
- 2.5 GHz clock speed
- GPU-NVIDIA GeForce 820M
- Frames per second input: 29.97 fps
- Frames per Second Output: 0.20 fps
- Video Format:.mov
- Size of the video: 4.14 MB
- COCO and self-created image dataset
- 5 classes have been trained.

Case II: Image specifications

- System Configuration—Intel i5 7th Generation (4Cores)
- 2.5 GHz clock speed

- GPU-NVIDIA GeForce 820M
- Input Image Size: 200-300 KB
- Training Time-0.6 seconds (SSD) 1.7 seconds (RCNN)
- Image Format:.JPG
- COCO and self-created image dataset
- Number of classes for which you have been trained: 5.

1. **For execution, hypotheses and limitations were made.**

- The gun is in the camera's line of sight and is fully or partially exposed to the camera.
- The backdrop light is sufficient to notice the ammo.
- To eliminate lag in ammo detection, a GPU with high-end compute power was used.
- This isn't a fully automated process. A person in charge will double-check every gun detection alert.

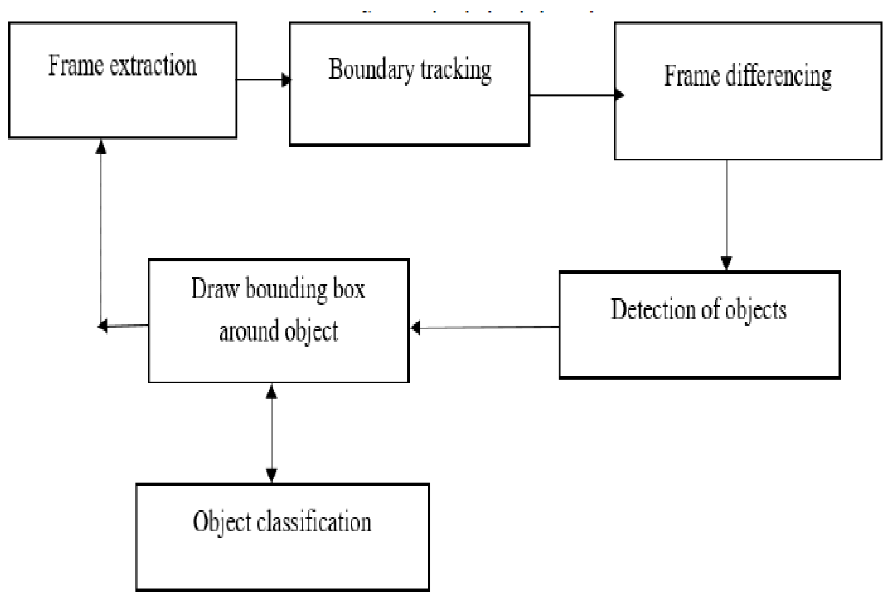


Fig 1: Methodology

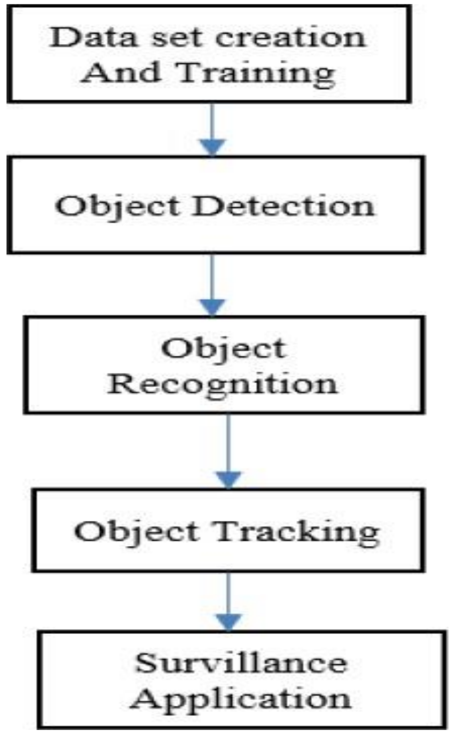


Fig.2. Detect ion and Tracking

FASTER R-CNN

Figures 3 and 4 show CNN layers and a faster RCNN architecture, respectively. It has two networks, one for generating region recommendations and the other for detecting objects.

It employs a selective search strategy to generate region-specific proposals. The RPN network ranks anchors, or "region boxes."

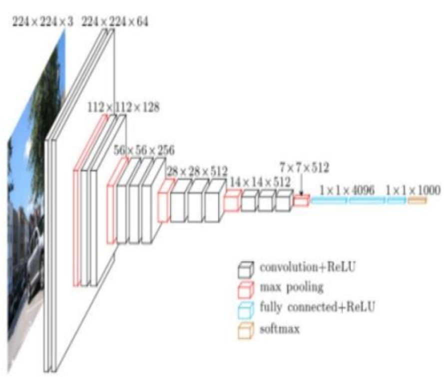


Fig 3. Layers in CNN Architecture [5]

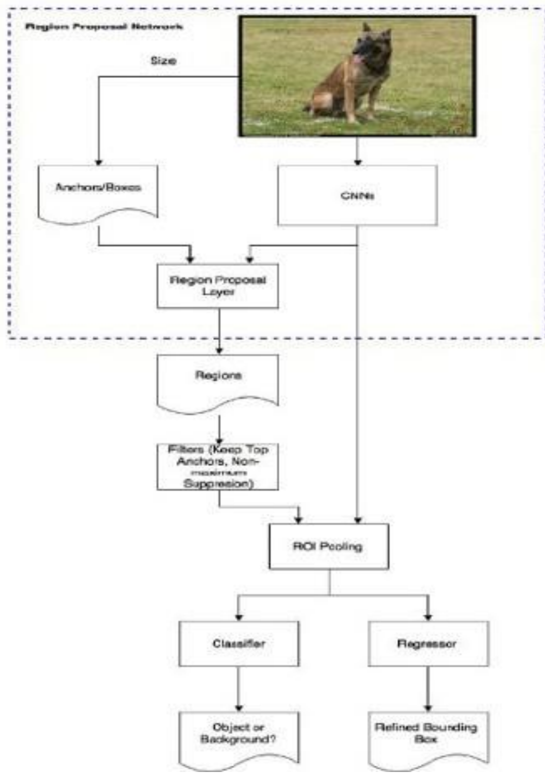
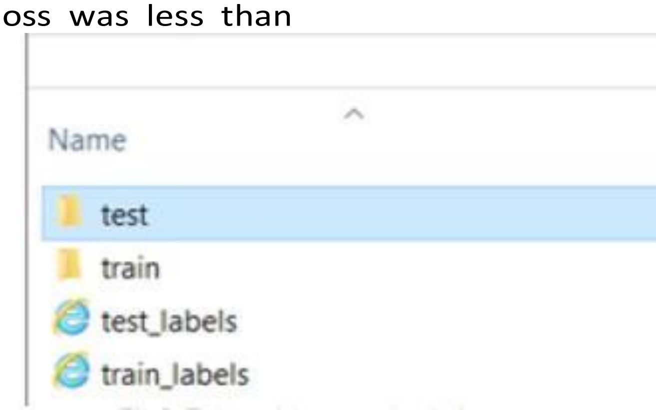


Fig 4. Faster R-CNN [5]

To download a large number of photographs at once, Fatkun Batch Image Downloader (a Chrome extension) is utilised. The images are then given descriptive captions. On the other hand, only 20% of the total number of photographs was actually utilised for testing. The resulting ammunition dataset was then trained using the Single Shot Detector (SSD) model, which went through 2669 iterations/steps to ensure that the loss was less than 0.05, boosting accuracy and precision. Figure 5 exhibits a collection of photographs from various types of testing and training. The picture in Figure 6 has been labelled. XML data is transformed into a CSV file by running this command in the Anaconda Prompt: python xml to csv.py. The resulting CSV files for the test and training datasets are shown in Figures 7 and 8, respectively.



ISSN No : 1006-7930

Fig.5. Folder with test and train images

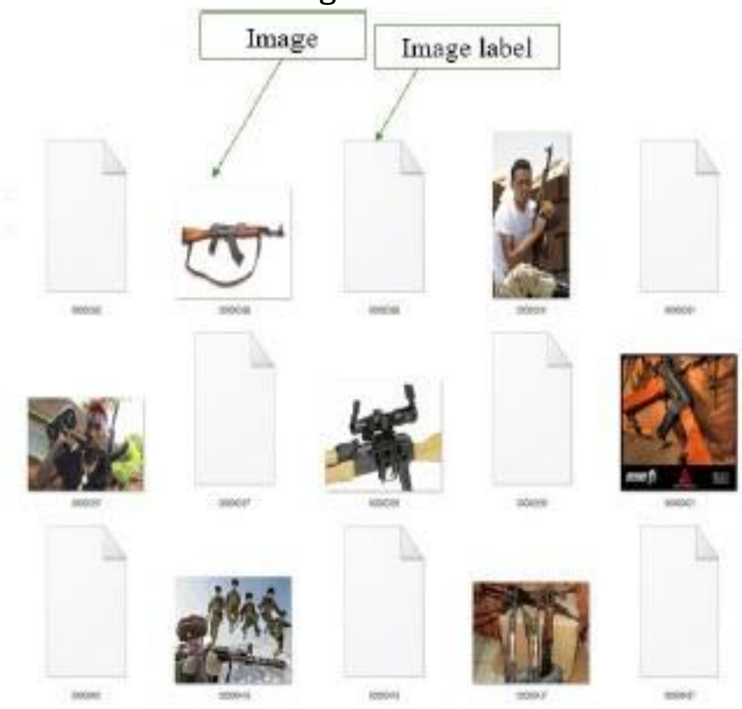
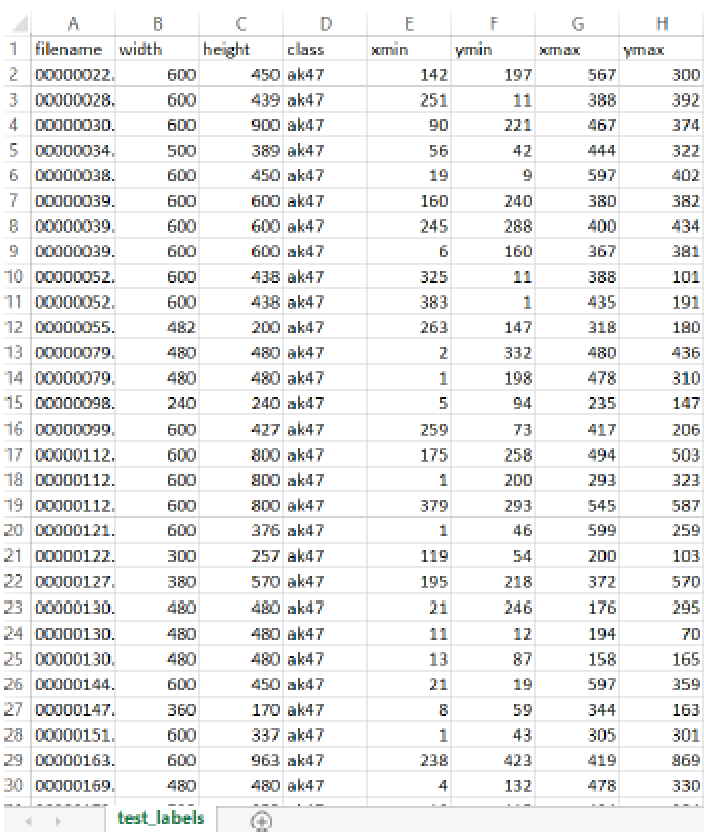


Fig.6. Image along with its label



	A	B	C	D	E	F	G	H
1	filename	width	height	class	xmin	ymin	xmax	ymax
2	00000022.	600	450	ak47	142	197	567	300
3	00000028.	600	439	ak47	251	11	388	392
4	00000030.	600	900	ak47	90	221	467	374
5	00000034.	500	389	ak47	56	42	444	322
6	00000038.	600	450	ak47	19	9	597	402
7	00000039.	600	600	ak47	160	240	380	382
8	00000039.	600	600	ak47	245	288	400	434
9	00000039.	600	600	ak47	6	160	367	381
10	00000052.	600	438	ak47	325	11	388	101
11	00000052.	600	438	ak47	383	1	435	191
12	00000055.	482	200	ak47	263	147	318	180
13	00000079.	480	480	ak47	2	332	480	436
14	00000079.	480	480	ak47	1	198	478	310
15	00000098.	240	240	ak47	5	94	235	147
16	00000099.	600	427	ak47	259	73	417	206
17	00000112.	600	800	ak47	175	258	494	503
18	00000112.	600	800	ak47	1	200	293	323
19	00000112.	600	800	ak47	379	293	545	587
20	00000121.	600	376	ak47	1	46	599	259
21	00000122.	300	257	ak47	119	54	200	103
22	00000127.	380	570	ak47	195	218	372	570
23	00000130.	480	480	ak47	21	246	176	295
24	00000130.	480	480	ak47	11	12	194	70
25	00000130.	480	480	ak47	13	87	158	165
26	00000144.	600	450	ak47	21	19	597	359
27	00000147.	360	170	ak47	8	59	344	163
28	00000151.	600	337	ak47	1	43	305	301
29	00000163.	600	963	ak47	238	423	419	869
30	00000169.	480	480	ak47	4	132	478	330

test_labels

Fig.7. CSV file of testing dataset

	A	B	C	D	E	F	G	H
1	filename	width	height	class	xmin	ymin	xmax	ymax
2	00000001.	600	346	ak47	48	25	562	297
3	00000003.	600	396	ak47	42	111	573	308
4	00000011.	320	212	ak47	22	67	306	148
5	00000014.	600	450	ak47	2	144	599	331
6	00000018.	600	400	ak47	170	137	490	397
7	00000020.	600	450	ak47	13	107	582	368
8	00000031.	221	160	ak47	16	7	212	153
9	00000040.	600	225	ak47	29	32	580	199
10	00000041.	600	600	ak47	5	228	597	375
11	00000051.	600	178	ak47	9	6	595	174
12	00000059.	599	448	ak47	89	31	404	186
13	00000059.	599	448	ak47	444	124	569	239
14	00000062.	600	337	ak47	4	115	553	226
15	00000066.	200	200	ak47	7	74	194	127
16	00000071.	600	428	ak47	153	104	550	387
17	00000072.	600	718	ak47	87	32	543	686
18	00000075.	600	216	ak47	11	22	587	195
19	00000076.	320	213	ak47	6	57	315	150
20	00000078.	600	450	ak47	20	33	554	415
21	00000083.	600	376	ak47	72	140	562	267
22	00000084.	600	800	ak47	244	126	348	778
23	00000084.	600	800	ak47	244	126	348	778
24	00000086.	600	300	ak47	13	83	584	255
25	00000088.	600	337	ak47	224	3	413	336
26	00000088.	600	337	ak47	224	3	413	336
27	00000092.	600	234	ak47	34	42	586	218
28	00000092.	600	234	ak47	34	42	586	218
29	00000096.	600	337	ak47	179	138	560	320
30	00000096.	600	337	ak47	179	138	560	320

train_labels

Fig.8. CSV files of Training dataset

SSD (Single Shot Detector)

In terms of accuracy and performance detection, the SSD algorithm has attained unprecedented heights. SSD shortens the process by doing away with the need for a regional proposal network. To make up for the lack of accuracy, the SSD makes use of a variety of technologies, including default boxes and multi-scale features. SSD can now match the faster R-accuracy of CNN while using lower quality pictures, resulting in significant performance gains. The COCO dataset has an average MAP score of 74% and a frame rate of 59 fps.

CONCLUSION

For weapon (gun) identification, the SSD and Faster RCNN algorithms are simulated using a pre-labeled and self-created picture dataset. Both methods are efficient and give good results, but their usage in real time demands a tradeoff between speed and accuracy. In terms of performance, the SSD method is quicker, at 0.736 frames per second. Faster RCNN, on the other hand, obtains a frame rate of 1.606 s/frame, which is sluggish when compared to SSD. The faster RCNN performs better in terms of accuracy, with an accuracy of 84.6 percent. When compared to the faster RCNN, the SSD has a lower accuracy of 73.8 percent.Due to its increased speed, SSD offered real-time detection, but faster RCNN gave better accuracy. It may also be used to train on bigger datasets using GPUs, high-end DSPs, and FPGA packages [16] [17].

REFERENCES

[1] Wei Liu et al., " SSD: Single Shot MultiBox Detector", EuropeanConference on Conputer Vision, Volume 169, pp 20-31 Sep. 2017.

[2] D. Erhan et al., "Scalable Object Detect ion Using Deep NeuralNetworks," *IEEE Conference on Computer Vision and PatternRecognition*(CVPR),2014.

[3] Ruben J Franklin et .al., "Anomaly Detect ion in Videos for VideoSurveillance Applicat ions Using Neural Networks," *InternationalConference on Inventive Systems and Control,2020.*

[4] H R Rohit et .al., " A Review of Art ificial Int elligence Met hods for DataScience and Data Analytics: Applicat ions and ResearchChallenges,"*2018 2nd International Conference on I-SMAC (IoT inSocial, Mobile, Analytics and Cloud),* 2018.

[5] Abhiraj Biswas et. al., " Classificat ion of Object s in Video Recordsusing Neural Network Framework," *International conference on SmartSystems and Inventive Technology,2018.*

[6] Pallavi Raj et. al.,"Simulat ion and Performance Analysis of FeatureExtract ion and Matching Algorithms for Image ProcessingApplicat ions" *IEEE International Conference on Intelligent SustainableSystems,2019.*

[7] Mohana et.al., " Simulation of Object Det ect ion Algorithms for VideoSurvillanceApplicat ions", *International Conference on I-SMAC (IoTin Social, Mobile, Analytics and Cloud),2018.*

[8] YojanChitkara et. al.,"Background Modelling techniques forforeground detect ion and Tracking using Gaussian Mixture model"*International Conference on Computing Methodologies andCommunication,2019.*

[9] Rubner et.al, " A metric for distributions with applications to imagedat abases", International Conference on Computer Vision,2016.[10] N. Jain et.al., "Performance Analysis of Object Detect ion and Tracking Algorithms for Traffic Surveillance Applicat ions using Neural Networks," *2019 Third International conference on I-SMAC (IoT inSocial, Mobile, Analytics and Cloud)*, 2019.

[11] A. Glowacz et.al., " Visual Det ect ion of Knives in Securit y Applicationsusing Active Appearance Model",Multimedia Tools Applicat ions,2015.

[12] S. P ankantiet .al.,"Robust abandoned object detect ion using region level analysis,"*International Conference on Image Processing*,2011.

[13] Ayush Jain et.al.," Survey on Edge Computing - Key Technology inRet ail Industry"

International Conference on Intelligent Computing andControl Systems,2019.

[14] Mohana et.al., Performance Evaluat ion of Background ModelingMet hods for Object Det ect ion and Tracking," *International Conferenceon Inventive Systems and Control,2020.*

[15] J. Wang et.al., "Det ecting stat ic object s in busy scenes", TechnicalReport TR99-1730, Department of Computer Science, CornellUniversity, 2014.

[16] V. P. Korakoppaet .al., "An area efficient FPGA implementation ofmoving object detect ion and face detect ion using adapt ive thresholdmethod," *International Conference on Recent Trends in Electronics,Information & Communication Technology,*2017.

[17] S. K. Mankaniet .al., "Real-t imeimplementat ion of object detect ion andt racking on DSP for video surveillance applications,"*International Conference on Recent Trends in Electronics, Information &Communication Technology,*2016.