International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 09 Issue: 04 | Apr 2022 www.irjet.net p-ISSN: 2395-0072

Improving AI surveillance using Edge Computing

Neha Saxena[1], Mr. Harsh Chauhan[2], Mr. Dhruv Khara[3], Mr. Rutvik Joshi[4]

[1] A.P., Dept. of Computer Engineering Universal College of Engineering, Mumbai university
[2,3,4] Dept. of Computer Engineering Universal College of Engineering, Mumbai university

---

Abstract - *Surveillance is a key part of security, assisting in the needs of the things to be protected. Oftentimes it happens that the human eye ignores essential key components while monitoring security throughout the campus which can result in an uncalled incident. Teachers often in classrooms are focused on teaching and assuring that students understand the concepts but eventually miss out on the overall gestures students indicate during the class. Using four powerful deep learning models namely person action detection, face recognition, landmarks-recognition and face re-identification recognition for smart classroom features combined with Mask-RCNN model for instance segmentation on object detection, particularly weapons, leverages the power of Artificial Intelligence to be deployed on the edge reducing the greater amount of resources required in case of cloud inferences. The real-Time monitoring system will help the users, particularly security personnel in understanding the on-ground situation of the campus. The system also warns on screen with a red border about the situation turning violent.*

Keywords: **Edge Computing, Mask-RCNN, Surveillance, Smart Classroom, Intel OpenVino toolkit.**

1. INTRODUCTION

Surveillance is an integral part of security and patrol. For the most part, the job entails extended periods of looking out for something undesirable to happen. Most of the criminal activities today take place using handheld arms, particularly guns, pistols and revolvers. The crime and social offense can be reduced by monitoring and identifying such activities of antisocial behavior which will allow law enforcement agencies to take appropriate action in the early stage. It is crucial that we do this, but also it is a very mundane task. Would not life be much simpler if there was something that could do the "watching and waiting" for us? With the advancements in technology over the past few years, Artificial Intelligence can help us automate the task of surveillance to a maximum level requiring only minimal effort of attention from the manual task force on the ground.

Deep learning is highly successful in machine learning across a variety of applications, including computer vision, natural language processing, and big data analysis. However, deep learning's high accuracy comes at the expense of high computational and memory requirements for both training and inference purposes. Training a deep learning model is space and computationally expensive due to the vast number of parameters that need to be iteratively refined over multiple time periods. High accuracy and high resource consumption are characteristics of deep learning models.

To complete the computational requirements of deep learning algorithms and models, a possible approach is to leverage cloud computing. To use cloud resources, data must be transferred from the data source location on the network edge to a central repository of servers which further increases the problems of latency, scalability and privacy hence edge computing is used as a viable solution.
Edge computing aims to solve the problems of latency, scalability and privacy by providing close computing proximity to end-to-end devices thus enabling real-time services and avoiding the dependency on centralized servers. Hence to avoid the computational heavy challenges of deep learning on the cloud or personal system, deep learning with edge computing is preferred over them.

2. EXISTING SYSTEM

2.1 Existing system

The existing approaches so far used are :

- Intelligent video surveillance
- Abnormal event detection
- Smart Classroom - An Intelligent Environment for Tele-education.

During our analysis of research papers, we found out that edge computing was not used for surveillance purposes. The existing system that we see requires a manual task force to actually monitor the input feed that is provided

through cameras. This feed is stored on cloud servers or physical servers causing storage issues every some interval of time.

The response to any alerting situation is considerably slow considering full automation which is a risk to the safety at the campus or the industry area where the system is installed. No system has yet used multiple and simultaneous deep learning models for surveillance purposes.

2.2 Project Scope

Using our proposed system, campuses can ensure they save more on resources and implement stricter methods of surveillance additionally allowing teachers to understand their students better and get an overview of how attentive students are during their learning.

The real-Time monitoring system will help the users, particularly security personnel in understanding the on-ground situation of the campus. The system also warns on screen with a red border about the situation turning violent.

3. PROPOSED SYSTEM

This chapter includes a brief description of the proposed system and explores the different modules involved along with the various models through which this system is understood and represented.

3.1 Analysis/Framework/Algorithm

- Face Detection Model[6] takes name as "input" with input image shape as [1x3x384x672] - An input image in the format [BxCxHxW], where: B - batch size, C - number of channels, H - image height, W - image width, Expected color order is BGR. The net outputs a blob with the shape: [1, 1, N, 7], where N is the number of detected bounding boxes will be the output of the image.

- Landmarks Regression Model[6] takes input with name as "data" and input image shape as [1x3x48x48] - An input image in the format [BxCxHxW], where :B - batch size, C - number of channels, H - image height, W - image width, The expected color order is BGR. The net outputs a blob with the shape: [1, 10], containing a row-vector of 10 floating-point values for five landmarks coordinates in the form (x0, y0, x1, y1, ..., x5, y5). All the coordinates are normalized to be in the range [0,1].

- Person Detection Model[6] takes input name: data with input shape as [1x3x544x992] - An input image in following format [1xCxHxW]. The expected channel order is BGR. 2) name: im_info, shape: [1x6] - An image information [544, 992, 992/frame_width, 544/frame_height, 992/frame_width, 544/frame_height]. The net outputs a "detection_ouput" blob with the shape: [1x1xNx7], where N is the number of detected pedestrians.

- Face Re-Identification Model[6] takes input name as "data '' , input shape as [1x3x128x128] - An input image in the format [BxCxHxW]. The net outputs a blob with the shape [1, 256, 1, 1], containing a row vector of 256 floating-point values. Outputs on different images are comparable in cosine distance.

- Mask RCNN - Using the instance segmentation dataset from Open Images Dataset V6 for weapon category objects, mask RCNN is trained for predicting and improving AI surveillance.

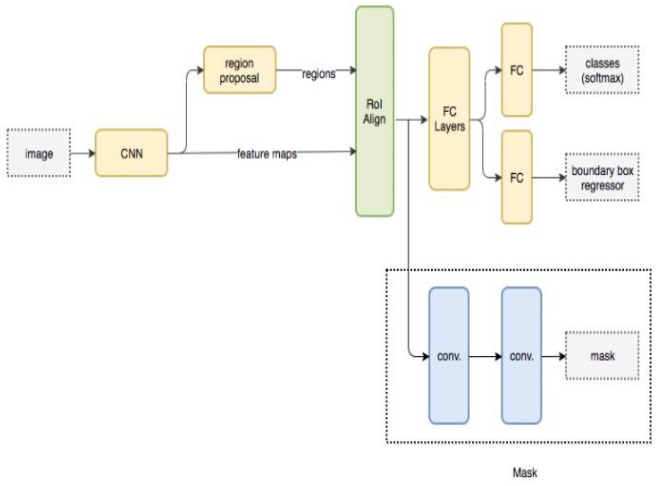

Fig -1: The architecture of Mask RCNN

The CNN structure used here is Feature Pyramid Network. The Feature Pyramid Network (FPN) is a feature extractor designed for such a pyramid concept with accuracy and speed in mind. The FPN network is capable of detecting very small objects. For proposed experimentation ResNet 101 with an FPN backbone is used. The backbone strides are taken as 4,8,16,32 and 64.

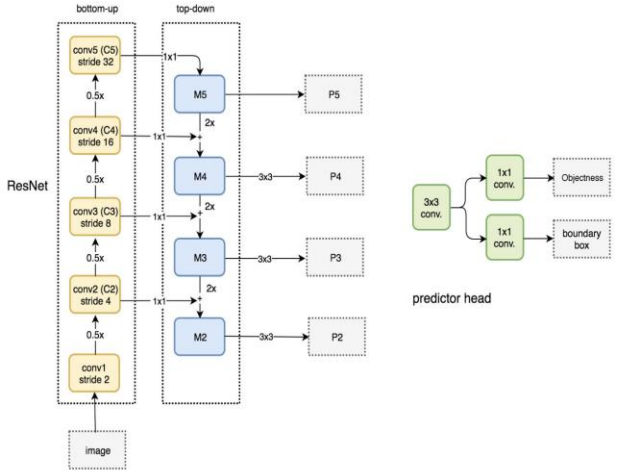

Fig -2: The Feature Pyramid Network with Resnet (ref-JonathanHui)

The FPN works as a feature detector and we extract multiple feature map layers and feed them to a Region Proposal Network or RPN for detecting objects. RPN applies 3 × 3 convolutions over the feature maps followed by separate 1 × 1 convolution for class predictions and boundary box regression. These 3 × 3 and 1 × 1 convolutional layers are called the RPN head. The same head is applied to all feature maps.

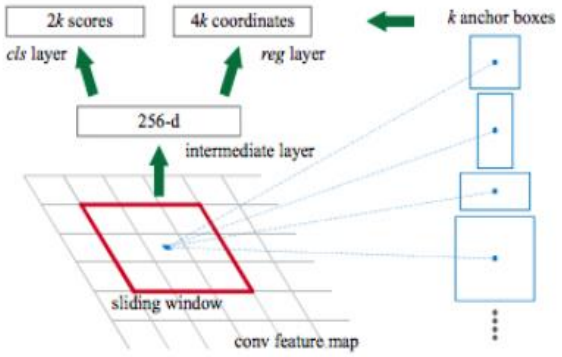

Fig -3: Anchor and sliding window in a vector form(ref-JonathanHui)

RPN having a classifier and a regressor. Here it is introduced the concept of anchors. Anchor is the central point of the sliding window. The length of the square anchor in pixels is taken as 32,64,128,256,512. The anchor ratios are taken as 0.5, 1 and 2 respectively. The ratio is defined by the width/height at each cell. For reduction in memory load, mini masks are used for high-resolution images. Mini masks reduce the masks to a smaller size to reduce the memory load.The height and width of the mini mask is taken as (56,56). Input images are resized to square form with a minimum image dimension of 800 and a maximum image dimension of 1024.

The Mask RCNN paper[2] uses 512 ROIs per image but often the RPN does not generate positive proposals to fill this and we keep a positive-negative ratio of 1:3. For ROI pooling the pool size is set as 7 and the maximum pool size as 14. The IOU thresholds are set at probabilities of 0.3 for minimum at 0.7 for maximum. For optimization of the loss function, we set the learning rate at 0.001 and the learning momentum of 0.9 with a weight decay regularization of 0.0001. For solving the exploding gradient problem we have defined a gradient clipping with the norm of 5.0. Every model is later passed onto a Model optimizer which generates a .xml file and a .bin file which are the model graph and weights file respectively. This is also known as Intermediate representation or IR. Furthermore, it is passed on to Inference Engine which is a set of C++ libraries with C and Python bindings providing a common API to deliver inference solutions on the platform of choice.

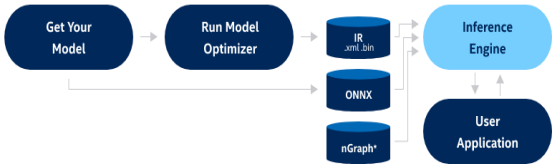

Fig -4: Intel OpenVino Toolkit internal working - source(docs.openvino.ai)

3.2 System Architecture

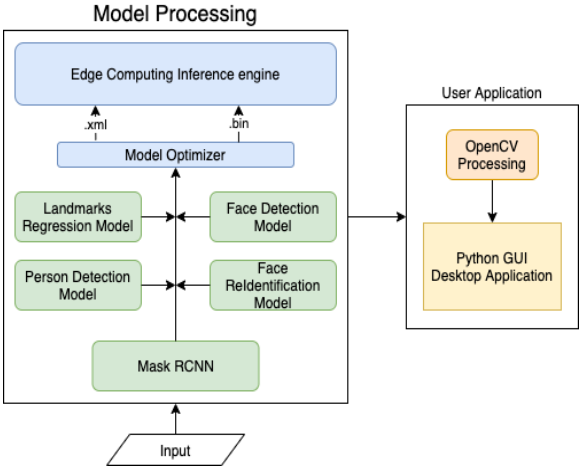

Fig -5: Proposed System Architecture

For the AI Surveillance system, an input feed from video cameras is processed through the established model pipeline as required for edge computing. The input feed is processed frame by frame and passes through the models for individual detection and action recognition for the frame. The models are passed through a model optimizer for generating the .xml and .bin files which are the model graph and weight files. Furthermore, the files are processed through the edge computing inference engine of Intel OpenVino for input inferences and forwarded to the user application module where the processed feed is displayed using OpenCV and for weapons being detected the border of the screen alerts in red.

Module Details

Model Processing - Once the input feed is processed through the model processing pipeline, the frames are resized according to the model requirements and are detected based on the model developed. Starting with the Face Detection model detects for the faces in the frames and is forwarded to the Landmarks regression model for landmark detection on faces. A student in the frame is then identified based on the available gallery of student faces using the Face ReIdentification model and then forwarded for entire frame analysis for weapon detection using mask RCNN. The model graphs and weights are processed using model optimizer and .xml and .bin files are generated which are forwarded to the edge computing inference engine.

User Application - The data processed through the Model Processing block is forwarded to display through the Python GUI using intermediary OpenCV processing steps. Depending on the identification of objects based on the MaskRCNN model the GUI will act accordingly. For Weapons detected, the display will show a red border indicating a warning and alert. For the Smart classroom, functionality outputs will be generated in a separate file.

4. EXPERIMENTAL RESULTS

Following the proposed algorithm pipeline and the system architecture, the assimilated results follow a significant improvisation depending on the system dependencies involved.

4.1 Smart Classroom Monitoring

Results obtained from the Intel OpenVino Edge computing toolkit while using it on an Intel i7 9th generation 9700K showcase the capabilities of handling 25 maximum frames per second with face identification, landmark detection, face reidentification, and action recognition models working simultaneously which over a cloud computing with 720P cameras has a processing speed of 10FPS[3] which is significantly low by 50%.

Fig -6: Smart classroom action recognition and face detection implementation

The data propagation round-trip time to edge servers and AWS a cloud computing service is 8 times higher than edge computing.[3]

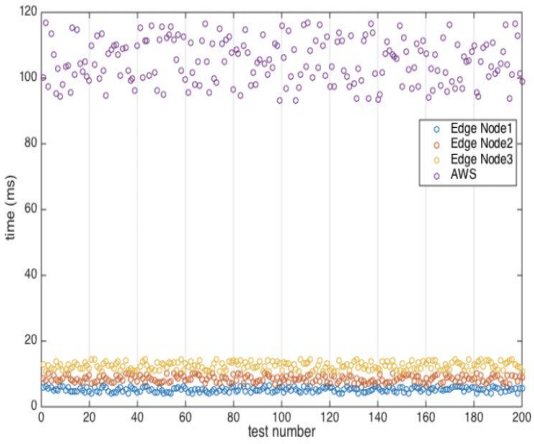

Fig -7: Wang, J., Pan, J., & Esposito, F. (2017). Elastic urban video surveillance system using edge computing. Data propagation round-trip time to edge servers and AWS.

4.2 Weapon detection using MaskRCNN

Fig -8: Weapon detection system initial implementation

Initial implementation of Mask RCNN is selected with the RPN anchor stride as 1. The concept of Non-Max Suppression is used to filter the RPN proposals. The RPN non-max suppression threshold is set at 0.7. For each image, we are training 256 anchors. After Non-max suppression, top 2000 ROIs for training and 1000 ROIs for generating inferences. Mask RCNN trained on pretrained COCO dataset using transfer learning algorithms brings a validation loss of 0.2294 which is selected for this proposed system.

4.3 Future Scope and Implementation

a) Detecting categories of weapons and identifying fake from real ones.

b) Detecting handheld guns using Pose and Human Key point Estimation algorithms.

c) Ensemble techniques for detection combining Instance segmentation with Human Keypoint estimation.

d) Creating a model API and pushing it to real-time deployment to a CCTV using a Raspberry PI.

5. CONCLUSION

In this proposed work we propounded and implemented a novel handheld gun classification and detection approach for surveillance additionally implementing a smart classroom monitoring system specifically for college surveillance purposes using edge computing. The system includes application of multiple models along with Mask RCNN giving us the optimal results. The system can identify the existence of numerous guns in real time and is robust across the variation in various factors of scale and rotation. Although, we assume that by implementing this method, the performance of the system will be refined and its real time processing essentials like complexity of space and time for processing can be diminished.

ACKNOWLEDGEMENT

We take this opportunity to express our deep sense of gratitude to Mrs. Neha Saxena for her continuous guidance and encouragement throughout the duration. It is because of her experience and wonderful knowledge, we can fulfill the requirement of completing within the stipulated time. We would also like to thank Dr. Jitendra Saturwar, Head of the computer engineering department and Mrs. Vishakha Shelke.

REFERENCES

[1] Zhang, Jolfaei, & Alazab, "A Face Emotion Recognition Method Using Convolutional Neural Network and Image Edge Computing", In-Edge Computing,(2019), IEEE

[2] Kaiming He, Georgia Gkioxari, Piotr Dollar, Ross Girshick ,"MASK R-CNN",In-Edge Computing,(2018), ICCV

[3] Jianyu Wang, Flavio Esposito, Jianli Pan,"Elastic urban video surveillance system using edge computing",In-Edge Computing,(2017), ACM

[4] Shivprasad Tavagad, Shivani Bhosale, Ajit Prakash Singh, Deepak Kumar , "Paper on Smart Surveillance System",In-Edge Computing,(2016), International Research Journal of Engineering and Technology

[5] Weikai Xie, Yuanchun Shi, Guanyou Xu, Dong Xie,"Smart Classroom - An Intelligent Environment for Tele-education",In-Edge Computing,(2001), Springer

[6] Intel's Pre-Trained Models or Public Pre-Trained Models OpenVINO™ Toolkit - Open Model Zoo repository, [Online] https://github.com/openvinotoolkit/open_model_zoo