# Object Detection in video streaming by CUDA – GPU using Machine Learning Techniques

Prajwal M, *M.Tech CSE*
*BMS Institute of Technology and Management,*
Bengaluru, India

Dr. Hemamalini B H
*Assosiate Professor*
*BMS Institute of Technology and Management,*
Bengaluru, India

Corresponding author:
Dr. Hemamalini B H

***Abstract*—This is a real-time project which hinder object tracking and recognition, an important sector of computer vision. Upon the rise of the interest in the advanced surveillance systems and other applications researchers developed new algorithms for the object detection and tracking. As the sample case we note person identification where the face is the subject of interest. The processing subsystem with elements responsible for detection and recognition, and the learning subsystem of the system are explained in the article in question. With this important information in mind, our goal is to help progress the field of real-time object recognition and tracking.**

***The actual object detection system developed in this paper is based on a real time Single Shot MultiBox Detector (SSD) architecture known as MobileNet-SSD. OpenCV has an exclusive feature called Deep Neural Network (DNN) which is used in implementing the present system for detecting an array of objects in both recorded and live videos. It is also able to identify things from 21 known categories such as everyday objects, automobiles and animals after the Caffe framework has been downloaded. Utilizing CUDA, the system boosts the performance up to the desired level of the GPU. This research demonstrates that the performance of object recognition is reasonably stable and accurate with an average of the FPS rates comparable with real-time systems such as surveillance and robotics. The offered approach can and should be enhanced and incorporated into more complex AI-based systems.***

Keywords— ***Object Detection, MobileNet-SSD, OpenCV, Real-Time, Deep Neural Network, CUDA, Caffe***

## I. INTRODUCTION

Object detection is important in computer vision systems. It can be used for many applications like video surveillance [1], object detection is able to provide valuable information for semantic understanding of images and videos and is related to many applications, including image classification [2]. It has great uses in areas such as self-driving vehicles, security cameras, and smart technology appliances. Nevertheless, real-time object detection is distinguished because the application can allow both the high speed and high accuracy of computations. Application of real-time object detection is however challenging than offline object detection. The system should not only be capable of recognizing and positioning the objects, but also do it in the least amount of time possible. This requirement put constraints on algorithm and hardware that need to be used for processing the data in real time.

This paper presents a system that solves the issues of real-time object detection using MobileNet-SSD architecture and OpenCV's DNN module. MobileNet-SSD is a deep learning for object detection which is optimized to be deployed on mobile and embedded platforms. This work provides a good tradeoff between computation time and detection rate that is ideal for real time use.

Its architecture is designed to support high-speed and accurate image and video processing thus meeting the model's goal. To improve the above aspects, we utilize CUDA based GPU acceleration where in FFT computations of GPU helps in increasing frame rates.

Through tuning of the MobileNet-SSD architecture and using Graphics Processing Unit, the system enables real-time object detection. It can be put to use in many areas that have need of object recognition in real-time, including self-driving cars, security systems, not to mention smart homes.

## II. LITERATURE SURVEY

The YOLO algorithmic program divides the important image into an N x N matrix and each network is supposed to detect the item located in this grid. Each matrix part or the grid cell then predicts bounding boxes and its several scores are

given as results [1]. High frame rate: Process frames at a rate of 45 fps, which is quicker than real-time (depending upon the size of the networks). The accuracy was 30.3% A much larger effect of IOU may be produced by a very low mistake in a very small box. A far larger impact on IOU may be caused by a very low mistake in a very small box.

The model relies on integrating the ImageAl deep learning libraries and You OnlyLook Once (YOLO-v3) object detection method with a DarkNet-53 architecture. The algorithm was trained using the TensorFlow framework to ensure accurate data processing [2]. The results show that the proposed method is superior to traditional object detection models, and the proposed model has an accuracy of increased by 78 %, and the model size is reduced by 26% improving the ability of generalization of the model and improving the performance of the model for complicated scenarios, which would make it more applicable to the embedded platform of the autonomous driving system.

The high accurate detection capability is desired, use Faster-RCNN with InceptionV2 [3]. For future research, the two models will be implemented as vision system of bomb disposal robot to detect improvised explosive devices. The SSD with MobileNetV1 has greater detection speed than Faster-RCNN with InceptionV2 when loaded to a video. But in terms of accurate detection, Faster-RCNN with InceptionV2 is more accurate. It shows that the SSD with MobileNetV1 has high speed detection but low accuracy compared with Faster-RCNN with InceptionV2 that has low speed but more accurate.

It is clarifying the ideas of model design and the limitations of deep learning method by overviewing the early object detection methods based on deep learning. It is clarifying the ideas of model design and the limitations of deep learning method by overviewing the early object detection methods based on deep learning [4]. The ROI pooling can help Fast R-CNN obtain the feature vector of fixed sizes, which is necessary to successfully connect with the full connection. The role of ROI pooling is just like the spatial pyramid pooling of SPP-net. The operation process of Fast R-CNN.

## III. PROBLEM FORMULATION

In the rapidly developing area of robotics, smart cameras, auto-mobiles and surveillance systems the requirement of real time identification of objects is exhilarating. In order to meet responsiveness real-time object detection requires fast and accurate identification of multiple objects within a live video stream while introducing minimum delays. However, the challenging tasks are often associated with obtaining accurate and fast object detection in real-time or on low-complexity devices. It is often crucial to detect objects in real time and, therefore, the method is widely used in robotics, cars without a driver, video surveillance.

Even if models as Faster R-CNN give good accuracy, issues of the increased number of computations may prevent their application in real-time use. Regardless, for the best real-time performance, even for such models like YOLO or SSD that are comparatively lightweight but offer a better compromise between speeds and precision, for some with less computing capabilities, it still would require optimization. The aim of this project is to construct the MobileNet-SSD, the approach that integrates the SSD into a lightweight MobileNet for building the real-time object detection system. Real time video feed should enable the system to name objects correctly and in the shortest time possible. CUDA, or GPU acceleration, refers to the utilizing of the graphics processing unit.

## IV. PROPOSED METHODOLOGY

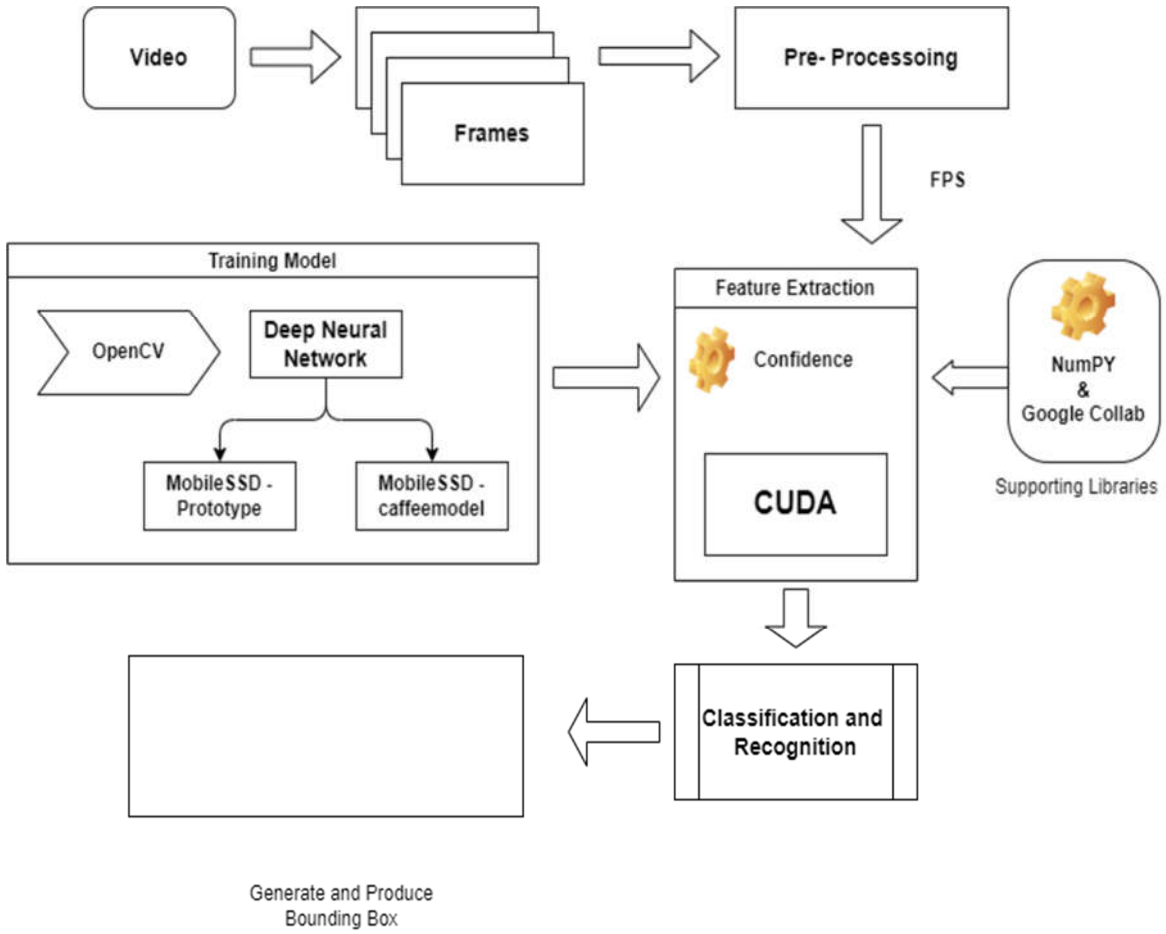


***Figure 1:*** *Block diagram of Methodology*

### *A. Data Collection*

The input video for a real-time object identification system is first prepared. This video can be a pre-set video or a live webcam feed. After reading the input video frame by frame, each frame is pre-processed and scaled to fit the needs of the object detection model. To help the model better understand the image data, the image is specifically shrunk to a fixed 300x300 resolution and the pixel values are normalized.

An exhaustive data collection endeavour will compile information from various sources, including: Firstly, the image turns into a structure 'blob,' or in other words, format that is comprehensible to the neural network. Next, the blob is passed into the MobileNet-SSD that has already been trained to identify 21 various types of objects such as people, cars, animals, etc. The picture passes through many layers within the model and other attributes that help in the identification of objects are obtained. These features are edges, forms and patterns.

### *B. Preprocessing of Dataset*

The photos are converted into blobs—a multidimensional array format that the MobileNet-SSD model can handle—after they have been shrunk and normalized. The blob has the image's height, width, batch size, and channels (such as Red, Green, and Blue). In order to extract and recognize features from the visual data, the neural network must first prepare it. The photos are converted into blobs—a multidimensional array format that the MobileNet-SSD model can handle—after they have been shrunk and normalized. The blob has the image's height, width, batch size, and channels (such as Red, Green, and Blue). In order to extract and recognize features from the visual data, the neural network must first prepare it.

The vocabulary is: [" background", "aero plane", "bicycle", "bird", "boat", "bottle", "bus", "car", "cat", "chair", "cow", "dining table", "dog", "horse", "motorbike", "person", "potted plant", "sheep", "sofa", "train", "tv",," monitor"] (size =21) It is further mapped with the indexes starting from 1 to 21.

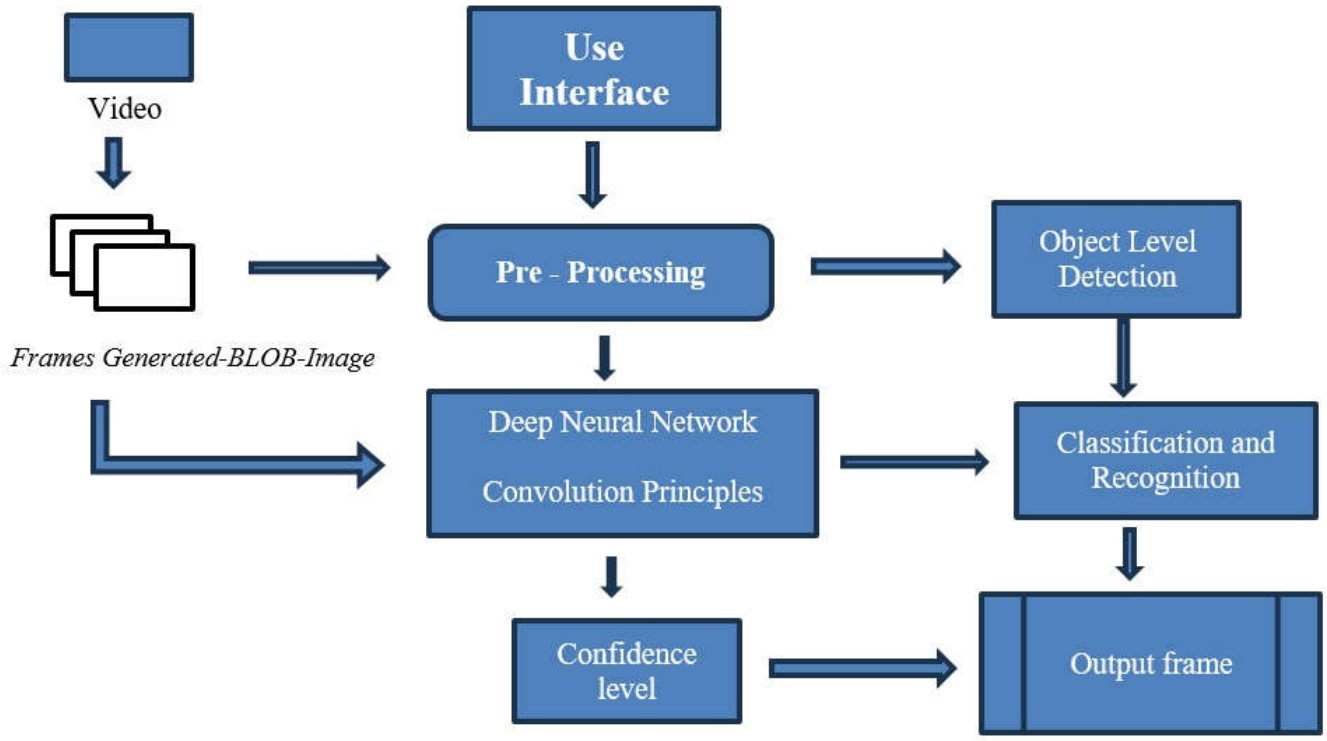


***Figure 2: Deep Learning Model for Object Detection***

### *C. Model Architecture*

MobileNet, a Convolutional Neural Network that has been conceptualised specifically for speed and indeed reliability. Unlike other convolutional layers, MobileNet employs depthwise separable convolutions to greatly decrease its complexity. This process involves splitting standard convolution operations into two simpler steps:

- **Depthwise Convolutions**: They are always employed on per channel fashion, meaning that one convolution operation is performed independently for each of the input channels (for example for each of the R, G, and B channels of an image). This step is to detect simple features edge level features.

- **Pointwise Convolutions**: A 1×11 times 11×1 convolution is applied to merge the outputs of the depth vise convolutions across channel. This step enables one to pick through more compound features. The model predicts a set of bounding boxes for objects. These predictions include:
  - The coordinates of the bounding box: (Xmin, Ymin, Xmax, Ymax) indicating the object's location.
  - A confidence score for each class, indicating how likely the object belongs to a specific class (e.g., person, car, etc.).

These bounding box predictions are made for different scales and aspect ratios, allowing the model to detect objects of various sizes effectively.

CUDA (Compute Unified Device Architecture) parallel computing platform and programming paradigm that gives application developers the ability to utilise the GPU (Graphics Processing Unit) for non-graphic computations. NVIDIA developed the GPUs that previously used to execute only graphical computations. But due to CUDA they can perform complex mathematical and computational operations much faster than CPU.

In this work, the improvement of the deep learning model is done with the help of CUDA that moves a number of computations such as convolution to the GPU for faster object detection. This enhances system productivity and allows identifying more frames per second - real-time object identification, necessary for such occupations as live video processing,

for example. To compare with the CPU-only processing, the system minimizes the delay by utilizing the GPU through CUDA to process many video frames and perform complex operations such as feature extraction and bounding box prediction.

### *D. **Training and Evaluation***

**a.** MobileNetSSD_deploy.prototxt: describes the structure of the network and does so by presenting its layers: activation, pooling, and convolution. Refers to the specification of input and output (e.g., the size of the input picture, the coordinates of the output detection). includes the architecture of the model containing layer types, padding, stride size, filter size, and other hyper parameters.

**b.** *MobileNetSSD_deploy.caffemodel:* is the parameter which occurred in the training stage of this model for every layer. Features that the model discovered using some dataset (for instance Pascal VOC) at the training step. The network uses these parameters to anticipate things based on incoming data (like picture frames). Role: This file is used during inference (detection) to plug in the actual learnt values that are used on a given input such as pixel intensities from an image. The input data analysis couldn't be done without this file and consequently, the model wouldn't be able to perform any detection. They can also see this as the "furnished" house which is all set to be moved into as the plan desired.

For a typical convolution operation:

- Let I1(x,y,c) represent the input to layer A.
- Let Wl(kx,ky,cin,cout) represent the convolutional weights (kernels), where Kx and Ky are the kernel dimensions, cin is the number of input channels, and cout is the number of output channels.
- Output feature map O(x,y,cout) is:

$$O_l(x', y', c_{\text{out}}) = \sum_{i=0}^{k_x-1} \sum_{j=0}^{k_y-1} \sum_{c=0}^{c_{\text{in}}-1} W_l(i, j, c, c_{\text{out}}) \cdot I_l(x + i, y + j, c)$$

Confidence is an estimation of the level of certainty that the object which a model has categorize belongs to a given class. It is given in the form of likelihood probability from 0 to 1. Having associated a bounding box to a detection, it is considered valid when its score of confidence is higher than a certain given value usually is 0.5.

$$c_i = \frac{e^{z_i}}{\sum_{j=1}^{C} e^{z_j}}$$

the model predicts a set of coordinates of axes around objects. These predictions include:
• The coordinates of the bounding box: Using the rectangular form of the coordinates, we have a couple of tuples containing the coordinates: (Xmin, Ymin, Xmax, Ymax) that points to the location of the object.
• A confidence score for each class of the object which means how much an object belongs to a specific class or e.g. person, car etc. (Xmin, Ymin, Xmax, Ymax) indicating the object's location.
• A confidence score for each class, indicating how likely the object belongs to a specific class (e.g., person, car, etc.).

### *E. Real Time Detection*

Now, with OpenCV for real-time sign language identification, live video feeds from a webcam or camera may be captured more easily. It dissected each frame in real time while regarding the visual input to ensure that the system was responsive. Combining the trained CNN model with OpenCV made the program predict sign language motions from the captured frames achieved during the training phase on the video si. For increasing uniformity and prediction accuracy, pre-processing demanded that every frame must be downsized and normalized.
Each time an object was recognised and the related Bounding Box is recognised and depicted on the screen, feedback was provided right away to the user. Real-time operation of the system increases the interest from the user and makes it applicable for various purposes such as assistive communication devices and teaching aids. Besides, there may be other option features like Live video using Tensor Flow to be included in the framework, to further improve the outcomes.

## V. RESULT

According to the collected reveals that GPU acceleration improves performance greatly and the real-time application can use the system. However, this is subject to existing complexity and quality of the input video stream in relation to the framed per second. By setting the confidence level to 0.5, the present model yields reliable detections but the confidence level can be adjusted depending on the specific application conditions.

The system is based on the assumption that the input space is comprised of item classes specified in advance, which proves to be disadvantageous when a system that is capable of finding more specific object is needed. May be part of future work aimed at expanding its scope of applicability. It could be retrained on unique data sets or enhanced with other detection work.

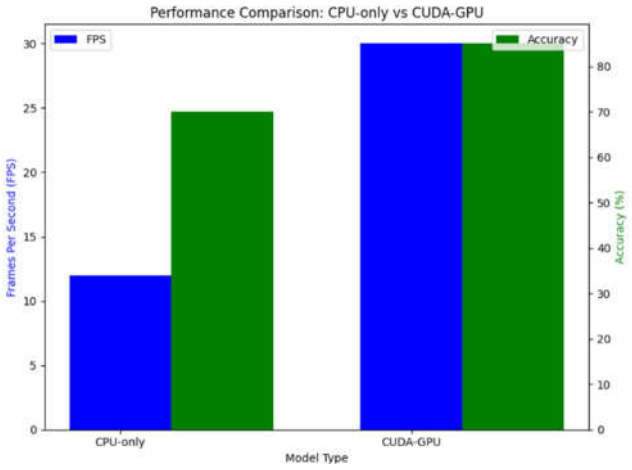


***Figure 3****: Graphical difference between CUDA and CPU*

The situations where the presence of GPU acceleration makes a difference is the case of achieving higher frame per second (25–30 fps with CUDA) compared to the processing by the CPU (10–12 fps). Outlier, low-confidence predictions are removed and those with the confidence level above the defined cut-off (0.5) are adequately emphasized. Its application is useful in real time applications, for instance in the self-driven vehicles; or in surveillance; since it achieves an optimal balance between the speed and the accuracy.

| Mode | CPU | CUDA - GPU |
|---|---|---|
| **Live Video** | 10 FPS | 25 FPS |
| **Recorded Video** | 12 FPS | 45 FPS |

***Figure 3:*** *Shows the Average Frames – per – Seconds the processor processing the given stream of video in the system.*

The model will use MobileNet-SSD model for object detection data by receiving image frames in a pre-processed form and extracting their features through a several layers of convolution as well as predict the class and the position of objects of the image based on some parameters which would be structured beforehand. This process happens continuously for each frame in real-time to develop bounding box with accuracy level the image been detected on top of the given resultant model shown in following (Figure: 4 and Figure: 5) in such a manner that all the component present in the resultant frame, will recognize the desirable object with the type of the object used and correct it when the confidence of the image is over 50% accurate with each frame with boundary box on it to generate the frame for processing.

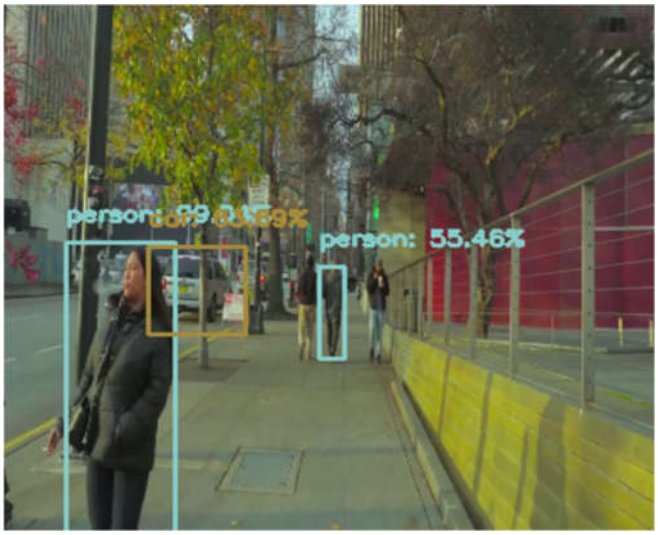


***Figure 4:*** *The Object recognition in the stream video with the accuracy of the clarity as in confidence with respect to model.*

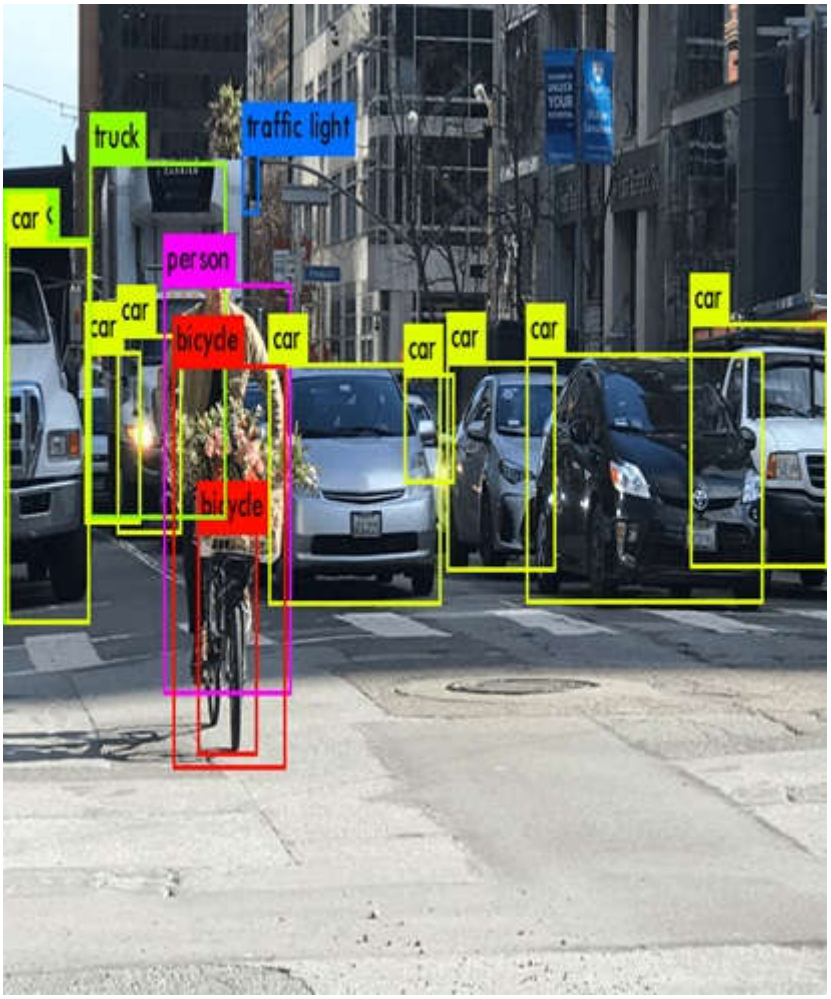


***Figure 5: Diagram of Model Architecture***

## VI. CONCLUSION AND FUTURE SCOPE

In MobileNet-SSD architecture, real-time object identification from SSD is integrated with agile feature extraction from MobileNet using depthwise separable convolution. The model has slim computational requirements, the speed and correctness of the object identification regardless of size and scale, as it performs bounding box regression and class prediction across multiple layers and can make the prediction with a single pass forward. Because of this, it is particularly useful for interactive applications where low latency is a requirement and the applications are run on low-end machines.

In OpenCV together with the utilization of CUDA-based framework for speeding up. Ultimate performance is maintained with live view and pre-recorded video streams to detect 21 object classes with better accuracy and frame rates. The basic performance of the system shows that it is suitable for surveillance, robotics and autonomous systems. Some possible future developments might concern the augmentation of the object classes and the optimising of the desired accuracy and performance with regard to the application in videos of higher resolution.

***Statements and Declarations***

- The authors did not receive support from any organization for the submitted work.
- No funding was received to assist with the preparation of this manuscript.
- No funding was received for conducting this study.
- No funds, grants, or other support was received.

**Financial interests:** The authors declare they have no financial interests.

## REFERENCES

[1] Cong Tang, Yunsong Feng, Xing Yang, Chao Zheng, Yuanpu Zhou, “The Object Detection Based on Deep Learning”, 4th International Conference on Information Science and Control Engineering, 2017, page 723-728.

[2] Pankaj Kumar Goswami, Garima Goswami, “A Comprehensive Review on Real Time Object Detection using Deep Learning Model”, Proceedings of the SMART–2022, IEEE Conference ID: 55829, 11th International Conference on System Modeling & Advancement in Research Trends, 16th–17th, December, 2022, College of Computing Sciences & Information Technology, Teerthanker Mahaveer University, Moradabad, India, ISBN: 978-1-6654-8734-4, page 1499-1502.

[3] “Object Detection and Recognition System Using Deep Learning Method”, Yashal Railkar, Aditi Nasikkar, Sakshi Pawar, Pranjal Patil , Rohini Pise, 2023 IEEE 8th International Conference for Convergence in Technology (I2CT), Pune, India. Apr 7-9, 2023, page 1-6.

[4] Ning Wang, Yinshan Jia, “Research on Vehicle Object Detection Based on Deep Learning”, 2023 4th International Conference on Computer Vision, Image and Deep Learning (CVIDL), 979-8-3503-2644-4/23/ IEEE, page 412-415.

[5] W. Ouyang, X. Wang, X. Zeng, et al, “Deepid-net: Deformable deep convolutional neural networks for object detection,” 2015 IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 2403-2412.

[6] P. M. Merlin, D. J. Farber, “A parallel mechanism for detecting curves in pictures”, IEEE Transactions on Computers, vol. C-24, Jan 1975, pp. 96-98.

[7] Y. Tian, P. Luo, X. Wang, et al, “Deep learning strong parts for pedestrian detection,” 2015 IEEE International Conference on Computer Vision, 2015, pp. 1904-1912.

[8] V. Gajjar, A. Gurnani and Y. Khandhediya, "Human Detection and Tracking for Video Surveillance: A Cognitive Science Approach," in 2017 IEEE International Conference on Computer Vision Workshops, 2017.