

A Novel approach for detection and Identification of Vehicles using Single Shot MultiBox Detector (SSD) and Real Time Analytics

Senthil Kumar A^{1*}, Selvaraj Kesavan², Ananda Kumar K S³, Nixon J S⁴, Senthil Kumar J⁵, Godhandaraman T⁶

¹Computer Science and Engineering, *Dayananda Sagar University Bengaluru*, India.

²Technical Architect, *DXC Technology*, Bengalore, India.

³Information Science Department, *Atria Institute of Technology*, Bengalore, India.

⁴Computer Science and Engineering, *Dayananda Sagar University*, Bengalore, India.

⁵Dept. of Computer Science & Engg *Sona College of Technology*, India

⁶Department of Data Science, *Dayananda Sagar University*, Bengalore, India.

Citation: Senthil Kumar A et al ,(2024), A Novel approach for detection and Identification of Vehicles using Single Shot MultiBox Detector (SSD) and Real Time Analytics, *Educational Administration: Theory And Practice*, 30(6), 2767 -2775, Doi: 10.53555/kuey.v30i6.5892

ARTICLE INFO

ABSTRACT

With growing vehicle density and traffic, manual detection and identification of vehicles is difficult. A lot of manpower is being used for regulating the vehicle and identifying the vehicle in case of violation, blacklisted vehicles, and vehicles of interest from agencies like transport, and law enforcement. Difficult to identify the vehicles from the place and track. It requires numerous resources. Agencies spend a lot of money and effort to identify the vehicles of interest. Manually checking the databases, finding the results, and taking action require skills and is time-consuming. It also leads to delays and difficulty in taking action in real-time. With the growth of computing infrastructure and the advent of artificial intelligence, many automated solutions are proposed to identify Vehicles. In this paper, the automated solution has been proposed to create a pipeline to detect, analyze, identify, and alert the stakeholders. The proposed solution is to identify the license plates of the vehicles from locations, detect the numbers, verify the vehicle details from the database in real time, and generate alerts in case of a red flag by sending messages about the location of the vehicle and the time stamp to law enforcement officers. The research proposes an approach to use deep learning based algorithms to identify the objects from visual inputs, analyze them in real-time, and generate notifications. The solution has been implemented using the Amazon Web Services platform and infrastructure and results are captured. The experimental result shows that default SSD boxes and multi-scale feature maps yield better accuracy than Faster R-CNN with lower-resolution images.

Keywords: Deep Learning, Object Detection, OCR, Real-time Analytics

I. INTRODUCTION

As the number of vehicles increases on the road, so does the situation arises to search for particular vehicles of interest by the law enforcement departments and other agencies.. The vehicle of interest could be any vehicle that is stolen, blacklisted due to various reasons, involved in unlawful activities and violations etc.. As of today, searching the vehicles is left to chance. Any sighting of a vehicle by police patrol or identified by civilians would be communicated to the authorities about the location. This is not an efficient way to solve crimes. Indian cities are among the top cities in the world that are equipped with traffic cameras to monitor the traffic. These cameras can be given the ability to track the vehicles and identify the number plate, read the numbers from the license plate and check the license plate details with the wanted vehicle information in real time or near real time and inform the law enforcement officer about the locality of the vehicle through much better communication like WhatsApp message or automated call to enable swift actions. The crux of the idea might have been impossible until a few years ago but with the advent of technology like Computer Vision, Real Time Analytics, Cloud Technologies, High speed internet, Cheaper Communication devices, Automated calls & WhatsApp it is possible in today's world. Computer Vision is an innovation that brings the ability to see and make decisions like humans do to machines and computers. It is a field of Artificial Intelligence that enables

computers to see the images & videos and identify objects by processing it. The application of this field in day to day life is immense and it can help us to tackle a lot of modern day challenges [1].

Object detection is the ability to detect and locate the object in the image using bounding boxes. Object detection heavily uses machine learning and deep learning algorithms to detect an object from an image or a video source. The detection can range from a single object to multiple objects from the same source. The accuracy of the detection depends on the algorithms and the database of images that has been used to learn the features. The performance and accuracy of the model in object detection is directly proportional to the variety of image databases used by the model to learn. Deep Learning models lay at the heart of object detection. Irrespective of the deep learning model used, the basic flow of creating an object detection model involves the following [2].

Creation of Source Data is the most difficult part of any data science project. We need to create a database with as many images / videos as possible. The variety of images in the database will help the model for better learning and prediction with high accuracy. Annotation of Images (Labeling) involves adding details to the image for the model to understand the features in the images. In this phase, we use tools to mark areas(bounding box) in the image to label them. The tool will generate xmls or jsons with the dimension (length, breath, height) of the bounding box in image along with the class name [3]. These xmls along with the images will be used by the models to learn about the class or features.

Selecting a deep learning model to use the source data and annotation to learn about the feature. Train the selected model on a training database and test the model accuracy in the test dataset. Tweak the model and tune the parameters based on the test results for higher accuracy. The main contributions of this paper include the following aspects.

Implement object detection solution using deep learning approach to detect the interested object Use OCR to detect and extract vehicle identity from number plate Implement the solution to stream the vehicle license plate information to a steaming engine and validate it against the Reference database. Business rules to check the vehicle against the defined list and generate alert notification.

In Single short multibox detector, The pre-computed, fixed-size bounding boxes known as priors are generated and closely resemble the distribution of the initial ground truth boxes. Actually, the way those priors are chosen ensures that their Intersection over Union ratio [10]. Equation 1 is the formula for deriving IoU .

Several improvements were made to this network to further enhance its ability to locate and categorize objects. For each of the b default bounding boxes per feature map cell and the c classes to classify, it is configured with two diagonally opposed points (x_1, y_1) and (x_2, y_2) ; on a given feature map of size $f = m * n$, SSD would compute $f * b * (4 + c)$ values for this feature map [11].

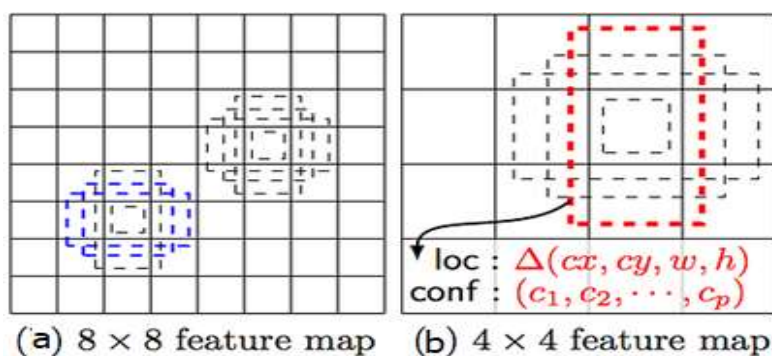



Fig. 1. SSD default matches with feature maps

SSD computes the location loss using smooth L1-Norm. It is still very effective and gives SSD more space because it does not strive to be "pixel perfect" in its bounding box prediction, despite being less accurate than L2-Norm. Since features maps, or the output of the convolutional blocks, represent the main features of the image at various scales, applying MultiBox to a number of feature maps raises the possibility that any object, no matter how big or small, will eventually be found, located, and classified correctly [12].

Having MultiBox on multiple layers leads in better detection as well because the detector runs on features at multiple resolutions. Having more default boxes yields more accurate detection, though there is a speed impact. Since the base VGG-16 network typically receives 80% of processing time, SSD performance might be further enhanced by a faster and more accurate network [9].

$$IoU = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$


II. RELATED WORK

One method for identifying the kind of target objects and locating them on a video frame is vehicle detection. Many object detection algorithms evolved over the years and the accuracy levels are improved due to optimized algorithms, powerful compute infrastructure and ability to process in real time [1]. Two stage object detection models divide the detection process into the region proposal and the classification stage. In the first step, the region of interest (ROI) is proposed on the source images in the database [2]. In the next step, the proposals are regressed and classified using various versions of convolutional neural networks (CNN). Region Based Convolutional Neural Network (R-CNN) utilize low level computer vision algorithms like Selective Search and Edgeboxes to generate regions of interest in the input images and use CNN to produce features and SVM to classify the features. Fast R-CNN- FAST R-CNN overcomes the challenges of the RCNN. In RCNN the regions of interest are created for images and depending on the size of the image, the number of ROI could grow exponentially. FAST RCNN uses ROI pooling layer to extract the most highlighted color which could wrap multiple ROI. We pass it to SVM for classification. The performance of this model is much better than RCNN Mask R-CNN is developed on top of Faster R-CNN. Faster R-CNN is a two stage process [3]. In stage 2 of Faster R-CNN, the ROI pooling is replaced by RoI align which helps to preserve the spatial information. The output of the ROI align layer is then fed into two convolution layers, which generates a mask for each RoI. This is predominantly used for object detection and segmentation [4]. Jahongir et. al., proposed a novel bounding box (Bbox)-based vehicle tracking algorithm with CNN-based classifier. A new vehicle dataset is prepared by annotating 7216 images with 123831 object patterns collected from highway videos. The classification accuracy of the Yolo-based vehicle detector was increased from 57% to 95.45% [6].

The single stage model removes the region of interest (RoI) extraction process and directly classifies and regresses the detection boxes. These single stage models are usually faster in inferring the features from the images. Single Shot Multibox Detected (SSD) - The SSD approach is based on a feed-forward convolutional network that produces a fixed-size collection of bounding boxes and scores for the presence of object class instances in those boxes, followed by a non-maximum suppression step to produce the final detections. You Only Look Once (YOLO) – YOLO is a single convolutional network simultaneously predicts multiple bounding boxes and the class probabilities for those boxes. YOLO trains on full images and directly optimizes detection performance. This way of detection is extremely fast [6].

A Kalman filter with two observers was employed by Shuming Du et al. to provide analytical redundancy for traffic state estimation. The outcomes show that the developed system is capable of real-time, accurate sensor fault detection and identification. Under fault-free and sensor-failure scenarios, mobility, safety, and sustainability all consistently improve [7]. CenterNet - this structure has an important advantage in that it replaces the classical NMS (Non Maximum Suppression) at the post process, with a much more elegant algorithm, that is natural to the CNN flow [8]. This mechanism enables a much faster inference.

III. PROPOSED SYSTEM

To detect, identify, analyze and notify the tagged vehicles, the end-to-end solution is proposed and the same is depicted in Fig 1. The solution comprises four different modules.

Object Detection - Using a Deep Learning Model to Detect the object in the photo or a video frame and locate a number plate from the vehicle.

Object Character Recognition – Use OCR to convert the characters in the license plate from image to alpha numeric format.

Real time streaming – Stream the vehicle license plate information to a steaming engine. The consumer, which would be an analytics engine, would pick the license plate information from the stream pipeline and validate it against the Reference database which contains the vehicle of interest information along with law officer contact information.

Analytics – the data from the stream pipeline can flow to a data lake which can be used by a visual tool like tableau to show near real time date on the vehicle that crossed a junction etc.,

A. Object Detection

A lot of time needs to be invested to select and fine tune the model from beginning for desired accuracy and speed. To address this challenge, many pre-trained models are available which can be used by the machine learning engineers to focus on solving the domain problem rather than creating the model and Tensorflow is picked up for object detection requirements.

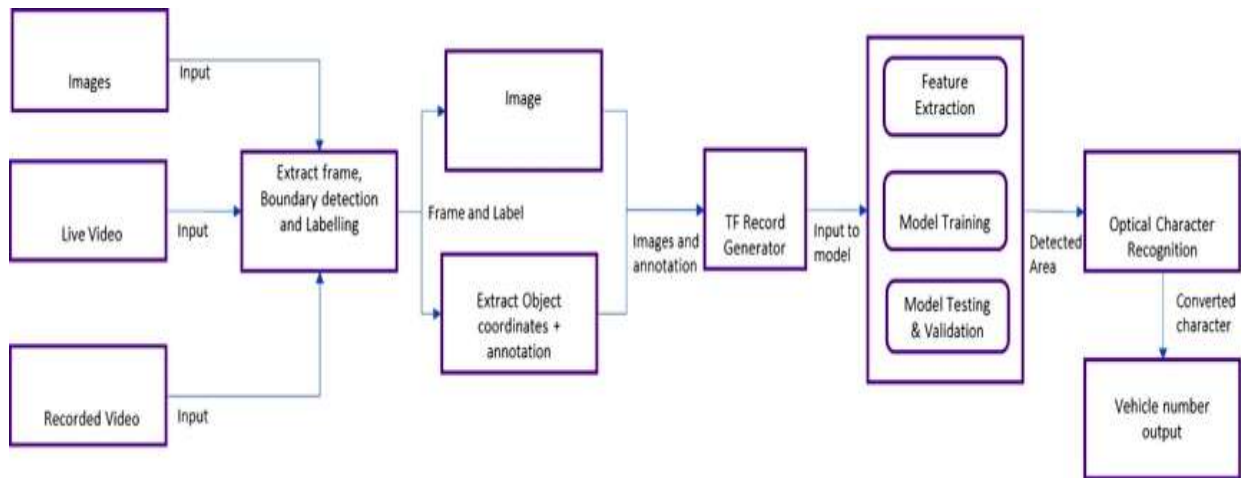


Fig. 1. Proposed System Architecture

Tensorflow is an open-source API that provides a set of comprehensive, composable and extensible low level API for high performance computation, primarily aimed at building machine learning (ML) models as well as authoring ML workflow tools and frameworks within the Tensorflow platform.

The TensorFlow Core APIs provide access to low level functionality within the TensorFlow ecosystem. This API provides more flexibility and control for building ML models, applications, and tools, compared to high-level APIs, such as Keras. Since the proposed solution is going to use object detection along with real time analytics, we need to choose a model that has balance between speed and mAP.

SSD MobileNet V2 FPNLite 320x320 model has been chosen for object detection and it has speed of 22 ms and mAP of 22.2. This model uses single stage object detection model Single Shot Multibox Detected (SSD) along with backbone network- Feature Pyramid Network (FPN) . Single shot detector used as VGG 16 or VGG19 network as a feature extractor. Then the custom convolution layers are added along with the convolution filters to make predictions. Feature Pyramid Network (FPN) is a feature extractor designed with feature pyramid concept to improve the accuracy and speed. It replaces the feature extractors like faster R-CNN and generates higher quality feature map pyramids.

The dimensions of the feature map are depicted in Fig 2.

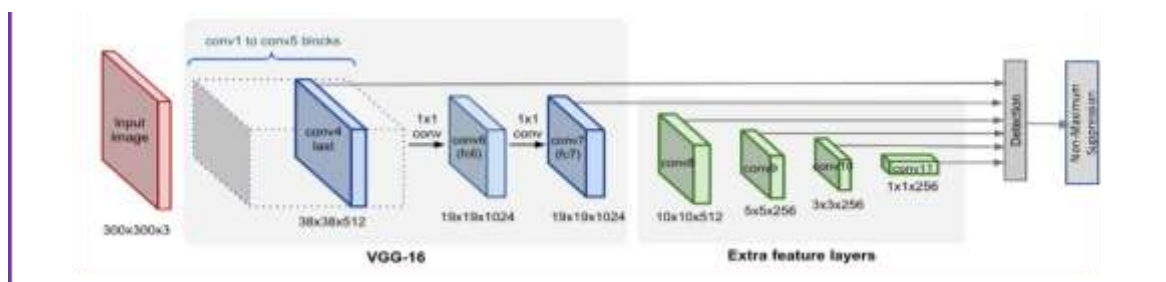


Fig.2. SSD Model

The chosen SSD MobileNet V2 FPNLite 320x320 model would use the combination of vehicle images along with its annotation around the license plate area to learn to detect the license plate in a vehicle. The public dataset from Kaggle is used in this solution. Annotations are created using LabelImg. It helps to select the area of interest and give a Label to it. In the below annotation, it is marked the area’s name as “License”. The XML gets generated for the image. The xmin, ymin, xmax, ymax marks the area of interest with the object name as “License”

```

<annotation>
<folder>PycharmProjects</folder>
<filename>Cars.jpg</filename>
<path>C:\Users\Senthil\Projects\Vehicle1.jpg</path>
<source>
<database>Unknown</database>
</source>
    
```

```

<size>
<width>1280</width>
<height>960</height>
<depth>3</depth>
</size>
<segmented>0</segmented>
<object>
<name>License</name>
<pose>Unspecified</pose>
<truncated>0</truncated>
<difficult>0</difficult>
<bndbox>
<xmin>515</xmin>
<ymin>666</ymin>
<xmax>788</xmax>
<ymax>729</ymax>
</bndbox>
</object>
</annotation>

```

Multiple datasets are used for the model training, and testing. The corresponding annotated xml files used to train and test the model. The model expects additional data in certain formats for it to train. Tensorflow object Detection API needs all the input data to be in a special format called TFRecord. A script is used to convert annotated XML to TFRecords.

Tensorflow object Detection APIs need the information on the classes created in the annotated xmls. Additional files should be created with information about the classes created in annotated xmls. The sample class information file will have the content as below

```
item { name:'licence' id:1 }
```

The config file captures the essential details such as path for the class information file, train, test record, loss function information and batch size. The config file also allows configuring the model to optimize the model to cater to the business need. Once the Model is set up, train the model to learn from the train data set. The data set split 80% to training and 20% for testing. Below is the screenshot (figure 3) of the model being trained with 410 cars dataset.

```

INFO:tensorflow:Step 9868 per-step time 0.187s
INFO:tensorflow:Step 9868 per-step time 0.187s
INFO:tensorflow:{"loss/classification_loss": 0.183845385,
"loss/localization_loss": 0.071832494,
"loss/regularization_loss": 0.122358816,
"loss/total_loss": 0.29628796,
"learning_rate": 0.073522527}
INFO:tensorflow:Step 9869 per-step time 0.187s
INFO:tensorflow:{"loss/classification_loss": 0.183845385,
"loss/localization_loss": 0.071832494,
"loss/regularization_loss": 0.122358816,
"loss/total_loss": 0.29628796,
"learning_rate": 0.073522527}
INFO:tensorflow:Step 9870 per-step time 0.187s
INFO:tensorflow:{"loss/classification_loss": 0.14588186,
"loss/localization_loss": 0.067637684,
"loss/regularization_loss": 0.122358816,
"loss/total_loss": 0.23532567,
"learning_rate": 0.073522527}
INFO:tensorflow:Step 9871 per-step time 0.187s
INFO:tensorflow:{"loss/classification_loss": 0.14588186,
"loss/localization_loss": 0.067637684,
"loss/regularization_loss": 0.122358816,
"loss/total_loss": 0.23532567,
"learning_rate": 0.073522527}
INFO:tensorflow:Step 9872 per-step time 0.187s
INFO:tensorflow:{"loss/classification_loss": 0.121367276,
"loss/localization_loss": 0.05837468,
"loss/regularization_loss": 0.12179182,
"loss/total_loss": 0.28144729,
"learning_rate": 0.073522527}
INFO:tensorflow:Step 9873 per-step time 0.187s
INFO:tensorflow:{"loss/classification_loss": 0.121367276,
"loss/localization_loss": 0.05837468,
"loss/regularization_loss": 0.12179182,
"loss/total_loss": 0.28144729,
"learning_rate": 0.073522527}

```

Fig. 3. SSD Training Output

Real Time Analytics Setup

The second part of the problem domain is to validate the license plate numbers identified and streamed from the model against the reference license plate maintained by the law enforcement officer [sample data] and to store the data in a database.

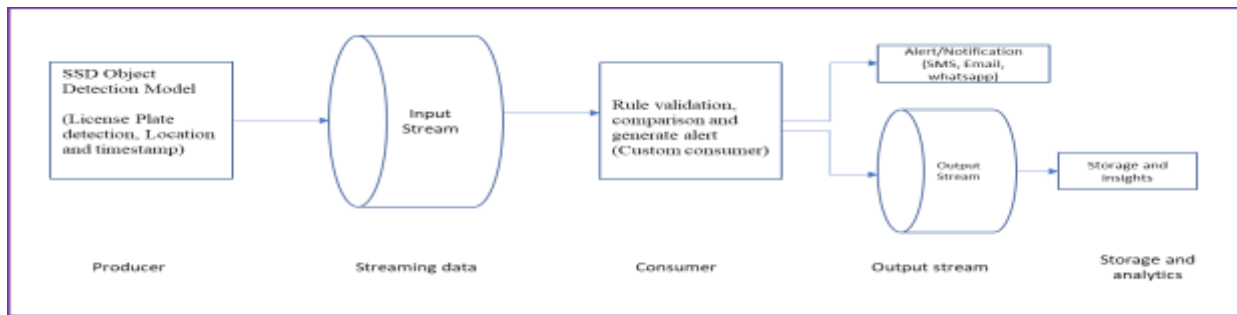


Fig. 4. Streaming Architecture

The streaming flow pipeline setup is depicted in Figure 4. AWS Kinesis is provisioned for a data streaming platform and multiple sources can be connected to it. It is a durable streaming service that aids the collection, processing and analysis of steaming data in real time. It is designed to allow you to get important information that is needed to make quicker decisions on time. Amazon kinesis Data Firehose is a service used to stream data from streams to destinations like Amazon simple storage Service (Amazon S3), Amazon Redshift, Amazon Open Search Service, splunk etc. Kinesis firehose does not need any code to be written or manage resources. Data producers need to configure to send data to a firehose and it automatically delivers the data to the destination that you specified. If the data needs to be transformed before delivering it, we can do it in Firehose.

Amazon Simple Storage Service (S3) is an object storage service offering industry - leading scalability, data availability, security and performance. Any amount of data can be stored for virtually any use case like data lakes, clout native applications and mobile apps. It is flexible to use any business intelligence and analytics platform. It gives options for the users to explore and manage data and faster to discover and share insights that can change the outcomes of the business. The solution implementation is described as below.

The License plate detector will detect the license plate from an image or video and add the geolocation information along with the time of detection and send it to the AWS Input data Stream. The sample output of detector is depicted in figure 5. Based on the input data volume, velocity, the number of shards can be dynamically managed based on the load.

The License plate information from the stream will be consumed by the custom consumer written in python. This consumer will get the reference data from S3 instance and compare near real time with the license plate information coming from the stream.

On successful comparison, a message is triggered to the phone number registered in the vehicle of interest database. Also a mail can be triggered to the registered email.

All the vehicle of interest information will be streamed to the S3 databases for further analytics.

Business intelligence tool connected with S3 data with the help of AWS Athena to generate a dashboard for further analytics.

B. Experiments

To detect the information in real time and generate output in real time, the real vehicle license plate information is fed as image to the algorithm and the data to the streaming pipeline and the vehicle has part of the Vehicle of Interest database stored in the S3 Database.



Fig. 5. License Plate Detection

The sample Vehicle of interest database is stored in my S3 bucket. It is in the form of a flat file. This reference data contains Vehicle of interest license plate number, an office phone number who is handling the case, the email id of the same office, the city and the station in which the complaint was made. As soon as the vehicle lincense plate is detected, it is sent to the input stream. The number of shards can be increased based on the load. When the data gets into the stream, it can load into any shard based on the load. Every item that enters the stream gets a sequence number for differentiation. It also carries the timestamp.



Fig. 6. WhatsApp Message

The output stream is pushed to S3 object storage and the same is captured in the Figure 6. The file was created from data push from vehicle-output stream.

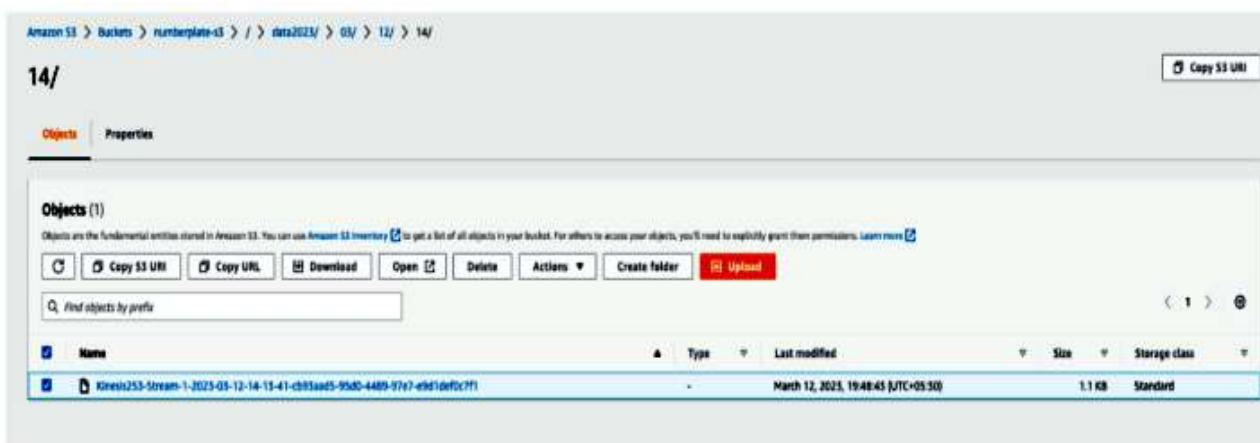


Fig. 7. Streaming to S3

Snapshot of the file from S3. All the vehicle information seen in vehicle-input-stream are available in the flat file (figure 8).

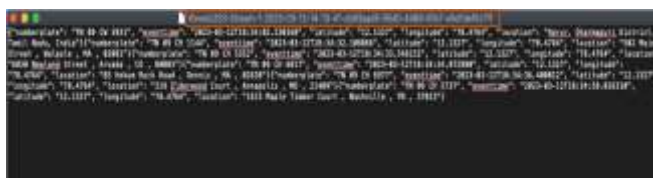


Fig. 8. Vehicle Input Stream Flat File

C. Performance Evaluation

The performance of the object detection model is measured by Mean Average Precision(mAP). The formula (Equation 1) to calculate the mean average precision is based on the confusion matrix, recall, precision and Intersection over union [14].

$$mAP = \frac{1}{N} \sum_{i=1}^N AP_i \dots\dots\dots (2)$$

Finding the Average Precision (AP) for each class and averaging it over several classes yields the mAP. The IoU threshold affects the AP metric. The below log gives the mAP from the model used for object detection. The precision of a prediction is the ability to identify true positives (TP) among all positive forecasts (TP + FP) [15].



Fig. 9. Model Performance

Table 1 shows the performance of the proposed model and the same has been compared with Faster R-CNN. The proposed model clearly edges in terms of accuracy and precision with less loss than the Faster R-CNN.

Table 1. Performance of Proposed Model

Performance Metrics	Faster R-CNN	Improved SSD
Mean Average Precision	0.499	0.518
Average recall	0.521	0.574
Total Loss	0.723	0.547

This model performance can be improved by optimizing the parameters and updating the configuration. However, the model detects the number plate and deciphers the Number plate information to a much better degree of accuracy. The performance of the data that gets into the stream created by the producer and message sent to the registered mobile number after validating it with the reference database is near real time since the consumer code is not deployed in the production server and is manually executed. However, if the consumer is deployed and in the continuous listening mode, the performance of the analytics can be improved by a huge margin. It is observed that the data from AWS firehose takes a little time to reach the Amazon S3 bucket. More experiments have to be done to measure the delay.

IV. CONCLUSION & FUTURE WORK

It is observed that the MobileNet 320x320 SSD model could detect the license plate information of the car fairly accurately and was able to successfully convert the license plate information to alpha number ostring for comparison with the reference data during real time analytics. The model was trained only with the car dataset. Hence the model was able to detect the license plate in the car only. With the variety of the vehicles running in the Indian road, It is needed to train the model with all kinds of vehicles to make the model more resilient and to increase the success of detecting the license plate of any vehicle in Indian roads. Plethora of analytics applications are available for real time streaming and near real time streaming. The streaming is mimicked for simple scenarios. With the availability of real data, we can create a streaming architecture that can scale to handle very high load. This dissertation concludes that we can send real time or near real time information on location of vehicle to interested persons. With more data, this model can be scaled. The Future scope of this idea has tremendous potential in the field of security. By optimizing the objection detection model for better accuracy and faster detection in video, this model can be installed in CCTV in the city junctions. The probability of identifying a vehicle of Interest that crosses the junction and getting notified to the nearby law enforcement officer is very high. If this setup can be scaled to multiple strategic key junctions in the city, any vehicle can be found within a very short lead time as it would be hard for any vehicle to miss the key junctions in the city. The technologies used for this project like high optical cameras, computing power needed for running the object model on a live feed and doing analytics on the real time data stream are evolving at a greater speed and are available for cheaper cost. Since this is an academic research, the model was trained on a limited dataset and real time analytics was tested on mimicked situations. However, this model can be improvised for production scale by training it with a variety of vehicles dataset and can configure it with a cost effective analytics platform. A model setup can be deployed on a trial basis on a junction to learn from its performance. Based on the performance, the model can be scaled to multiple junctions to serve as a tool for the law enforcement officers to solve crime.

REFERENCES

- [1] Silva, S.M., Jung, C.R.: Real-time brazilian license plate detection and recognition using deep convolutional neural networks. In: 2017 30th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI). pp. 55–62, 2017.
 - [2] Li Kang, Zhiwei Lu, Lingyu Meng, Zhijian Gao, “YOLO-FA: Type-1 fuzzy attention based YOLO detector for vehicle detection”, *Expert Systems with Applications*, vol. 237, pp. 121209, ISSN 0957-4174, 2023.
 - [3] Jahongir Azimjonov, Ahmet Özmen, “A real-time vehicle detection and a novel vehicle tracking systems for estimating and monitoring traffic flow on highways”, *Advanced Engineering Informatics*, vol. 50, pp. 101393, ISSN 1474-0346, 2021.
 - [4] Shuming Du, Saiedeh Razavi, “Fault-tolerant control of variable speed limits for freeway work zone using likelihood estimation”, *Advanced Engineering Informatics*, Vol. 45, pp. 101133, ISSN 1474-0346, 2020.
 - [5] Bulan, O., Kozitsky, V., Ramesh, P., Shreve, M.: Segmentation- and Annotation-Free License Plate Recognition With Deep Localization and Failure Identification. *IEEE Transactions on Intelligent Transportation Systems* 18(9), 2351–2363 (sep2017).
 - [6] Prashant Deshmukh, G.S.R. Satyanarayana, Sudhan Majhi, Upendra Kumar Sahoo, Santos Kumar Das, “Swin transformer based vehicle detection in undisciplined traffic environment”, *Expert Systems with Applications*, vol. 213, pp. 118992, ISSN 0957-4174, 2023.
 - [7] Selmi, Z., Ben Halima, M., Alimi, A.M.: Deep Learning System for Automatic License Plate Detection and Recognition. In: 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR). pp. 1132–1138. IEEE (nov 2017).
 - [8] Kekevi, Ugur & Aydin, Ahmet. (2022).: Real-Time Big Data Processing and Analytics: Concepts, Technologies, and Domains. 7. 111-123. 10.53070/bbd.1204112.
 - [9] Lubna Aziz, Md. Sah Bin Haji Salam FC, Sara Ayub, “Multi-level refinement enriched feature pyramid network for object detection”, *Image and Vision Computing*, vol. 115, pp. 104287, ISSN 0262-8856, 2021.
 - [10] Delmar Kurpiel, F., Minetto, R., Nassu, B.T.: Convolutional neural networks for license plate detection in images. In: 2017 IEEE International Conference on Image Processing (ICIP). pp. 3395–3399. IEEE (sep 2017).
 - [11] Du, S., Ibrahim, M., Shehata, M., Badawy, W.: Automatic License Plate Recognition (ALPR): A State-of-the-Art Review. *IEEE Transactions on Circuits and Systems for Video Technology* 23(2), 311–325 (feb 2013).
 - [12] Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You Only Look Once: Unified, Real-Time Object Detection. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 779–788. IEEE (jun 2016). <https://doi.org/10.1109/CVPR.2016.91>.
 - [13] Milosevic, Zoran & Chen, Weisi & Berry, A. & Rabhi, Fethi. (2016). Real-Time Analytics. 10.1016/B978-0-12-805394-2.00002-7.
 - [14] Alberto Rizzoli : The Ultimate Guide to Object Detection. url : <https://www.v7labs.com/blog/object-detection-guide>
- Anton Morgunov : How to Train Your Own Object Detector Using TensorFlow Object Detection API url : <https://neptune.ai/blog/how-to-train-your-own-object-detector-using-tensorflow-object-detection-api>