Vehicle Accident Detection and Alerting System on IoT System

Aman Joshi (IIT2018042), Anish Gir Gusai (IIT2018044), Avishek (IIT2018051),
Nagendra Singh Tomar (IIT2018057), Laxman Goliya (IIT2018066)
VI Semester B.Tech, Information Technology,
Indian Institute of Information Technology, Allahabad, India
Project Guide: Dr. Triloki Pant

***Abstract*—On Road accidents have become so rife that the fatalities due to these accidents have seen a steep rise. Accidents are a menace not only to human life, but resources, and it creates disruption in the normal traffic flow.The more concerning characteristic of the on road accidents is the overall damage rate that ensues these accidents. Now, when Machine Learning has taken over, the previously complex problems have become feasible, and the real life applications of these artificial ML models have been very promising. In this paper, We propose a learning Model that learns over an image dataset, thereby classifying never before seen images and data. With this model, we aim to predict and detect real-time collisions and accidents through a security camera planted along the periphery of the highways. We aim to classify these real-time accidents based on the level of damage. We will make use of the Artificial Neural Network to train the model to learn the similarities among images and accident data.**

***Index Terms*—Detection, ReLU, CNN, Faster R-CNN, Convolutional Layer, YOLO, Context Augmentation**

I. Introduction

Artificial Neural Network is a computational system, wherein a number of layers of neurons are associated together, or are stacked one over the other in such a way that the output of one layer subsequently becomes the input of the next layer, until the final layer associates the input to a desired output. The prediction may not always be as expected, and in such a case, we apply a correction function that minimizes the error. This error correction is achieved using a back-propagation technique.
Neural networks use basis functions that follow the same form as, so that each basis function is itself a nonlinear function of a linear combination of the inputs where the coefficients in the linear combination are adaptive parameters. This leads to the basic neural network model, which can be described as a series of functional transformations.
The Linear Classification Models don't work for complex featureset and dataset. This is due to the non-linearity and complex structure of data such as images, videos, audio, etc.The fetures that needs to be taken into account is not specified or there are simply too many features present in the data. We then use complex neural Models that emulate Human Like intelligence through activation function implemented on neurons. The system models the learning process on dataset by minimizing the overall error due to the incorrect prediction. System like the Neural Network work by combining various elements and features of the dataset and processing these combinations in a number of layers. while most of the models implement a similar procedure, some exception may exist.
Ariticial Neural Network could be categorized under the learning procedure they tend to adopt.

A. Motivation

Millions of people, all over the globe lose their lives to road accidents annually. Statistics implies that 30% of dead could be saved if life saving treatment is provided within the crucial hour . There has been profound efforts undertaken by various NGOs and voluntary organisations to mitigate fatality rate by dispensing the necessary intime services. This model could come in handy to alert the intime service carriers.

B. Objective

- This model increases the efficiency of existing models.
- Accident Classification based on the level of damage.
- Another important objective is augmenting the existing Datasets in order to create a more generic dataset for training models on accident detection and classification.

II. LITERATURE REVIEW

A. A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects (ZewenLi, WenjieYang, Shouheng Peng, Fan Liu, Member, IEEE)

A number of CNNs have been proposed. As we slowly begin to understand the shortcomings with a particular model, we try to minimize the effect of those shortcomings, which in turn, gives rise to newer extensions to the already established methods, creating methods that eke out efficiency and complexity. The advantages of the convolutional neural network have been discussed at length in the above-mentioned Literarture. Some of these ideas are related to down-sampling (or dimensionality reduction), weight sharing and local connection. The paper gives the detailed survey of the CNN including applications, classic networks, building blocks, related functions and prospects. Construction of a CNN-based model from scratch has been taken up and discussed quite efficiently. Furthermore, some of the guidelines in devising a network aiming to improve speed and accuracy, as we know that building a deep neural network can improve the ability of the neural network to handle the complex tasks, such as low rank approximation, dimension reduction, faster computing speed. Certain optimizers have been discussed which give us an idea of selecting the appropriate functions. Various loss and activation functions have been discussed in a very alluring way, mentioning the advantages and shortcomings of every function that is used. The activation functions include sigmoid, tanh and ReLU. It also discusses certain other forms of the commonly used Non-linear Function ReLU such as PReLU and leaky ReLU. the paper also postulates various ways to choose and optimize error and loss functions. To name a few, MSE (Mean Square Error), MAE (Mean Absolute Error), Cross Entropy, Contrastive Error. Back- propagation has also been discussed sufficiently. In this paper, we saw some typical application of the CNN and the different dimensions used for the various problems. Also this paper gives us the challenges faced by the CNN such as- poor crowded scene result, lack of equivariance, low generalization, which makes CNN hard to handle.

B. Accident Forecasting in CCTV Traffic Camera Videos (Ankit Shah, Jean Baptiste Lamare, Tuan Nguyen Anh, Alexander Hauptmann)

This paper has proposed the dataset CADP - Car Accident Detection and Prediction (CADP) dataset for Traffic Camera based accident forecasting. The discussion begins with the need for a new dataset,and thereby it establishes the basis of a new dataset. The advantages and limitations of other datasets are taken up at a greater length. The paper gives a detailed description about the CADP dataset right from the development to the results and its future scopes for application in different domains. Later on, the authors have also shown the results of classification using Faster R-CNN along with context mining and augmented context mining. Comparison is made between SSD and Faster R-CNN based upon experimental results on a new trainval set especially sampled for cross validation. They presented the results of state-of-theart object detection and accident forecasting models on their dataset, highlighted the strengths and weaknesses of baseline models, and outperformed the initial results by adding context mining or augmented context mining. The results showed that augmented context mining does not improve the score obtained with a gradual context mining for object detection.

C. A Vision-Based Video Crash Detection Framework for Mixed Traffic Flow Environment Considering Low-Visibility Condition (Chen Wang ,Yulu Dai ,Wei Zhou ,and Yifei Geng)

There has been various research conducted with the aim of developing vision-based system to detect traffic crashes. The paper mentions three categories of study:- (1) modelling the traffic flow (2) modelling of vehicle interactions, and (3) Analysis of activities. Analysing patterns of vehicle flow from previously seen data and models, it becomes important that real-time is consistent with the flow model, otherwise, the classification yields unsatisfactory results. Another mathod is to detect speed change through high sensitivity cameras. The third category mainly depends on using tracker facilities to monitor features such as speed, Force, accelration, etc. Another practical limitation of the model is the environmental effects that hampers the efficiency of the model(foggy conditions, Torrential downpour and insufficient-illumination). Various Image enhancing features could mitigate these effects. But, then, the complexity of the image-processing model increases manifold. The camera-based detection suffers mainly due to the computational complexity of the model, which brings down the latency in real-time implementation. Choosing a good leaning and classification algorithm is also the key to a better Model.

D. Accident Detection Using Mask R-CNN (Akshit Diwan, Vandit Gupta1, Chaitanya Chadha)

The main objective of this research paper is to detect road accidents. In this research paper the mask R-CNN is used for accident detection of vehicles. The methodology proposed in this research paper is divided in two parts, Transfer learning and Mask R-CNN. Transfer learning is the methodology in which a model that is being trained over one task is repurposed on a second related task. As a result it helps in improving the performance while modeling the second task. In this paper Mask R-CNN is implemented in two stages: first region proposals are generated based on the presence of an object, and Regional proposals network is used for implementing the first stage. Second, the object class predictions are performed; this is implemented by using the binary mask classifier that generates masks for the object class and after this collision is detected by calculating the intersection over union concept which gives the measure of intersection between two bounding boxes of two different vehicles and if the value is higher than the threshold hence collision has occurred.

E. A New Video-Based Crash Detection Method: Balancing Speed and Accuracy Using a Feature Fusion Deep Learning Framework (Zhenbo Lu, Wei Zhou, Shixiang Zhang, Chen Wang)

A New Video-Based Crash Detection Method: Balancing Speed and Accuracy Using a Feature Fusion Deep Learning Framework. In this research a new methodology has been discussed for crash detection. The main objective of this paper is to fill the gap between speed and accuracy for detecting crashes.
Methodology: To capture the appearance features of crash images ResNet with attention modules was developed in order to boost the models speed as compared to other video based crash detection models because ResNet is much faster than conventional convolution neural networks. Attention module directs the ResNet to only focus on localized appearance features and ignores inappropriate information. Which helps in improving the performance of the model. Conv-LSTM is used to capture motion features of crashes as it captures motion features better than LSTM. Hence the model performs better in terms of speed and accuracy by achieving the overall accuracy of 87.78% and fast detection speed greater than 30. So the proposed model maintains and fills the gap between speed.

F. Computer Vision based Accident Detection for Autonomous Vehicles (Dhananjai Chand , Savyasachi Gupta , Ilaiah Kavati)

Various Deep Learning and sensor-based models have been created to distinguish possible mishaps with a self-ruling vehicle. Be that as it may, a self-driving vehicle should have the option to distinguish mishaps between other vehicles in its way. Also, self-ruling vehicles can report the nearby specialists when they distinguish an already happened mishap which can accelerate the way toward alarming the clinical helpers and taking care of the episodes post-mishaps (many human setbacks happen because of postponement in announcing mishap cases in a timely manner creating further setbacks for getting clinical help.) In this paper, a novel supportive network is proposed for self-driving vehicles that recognizes vehicular mishaps through a camera installed in them.. The framework uses the Mask R-CNN structure for vehicle's identification and a centroid following algorithm to follow the identified vehicle. Also, the structure ascertains different measurable factors like speed, speed increase, and direction to decide if a mishap has happened between any of the followed vehicles. The system has been tried on a custom dataset of dashcam film and accomplishes a high mishap discovery rate while keeping a low false alert rate.

G. An Improvement of Traffic Incident Recognition by Deep Convolutional Neural Network (Hoai Nam Vu, Ngoc Hung Dang)

Due to the highly increasing number of vehicles, traffic problems have become more and more terrible over the world. According to the WHO report, there are about 1.35 million deaths and 20-50 million injuries globally every year as a result of vehicle accidents. Therefore, it is important to develop an efficient accident detection system, which can reduce both the number of deaths and injuries. This paper proposes a methodology that combines CNN object classifier and Faster R-CNN object detector, which can efficiently detect traffic incidents under real-world conditions. To detect traffic events detection and recognition on separate lanes this method used recent deep learning models. We experimented on a real-world dataset, and the results of the experiment prove that this method effectively located the accident while ensuring a real-time scenario of the system.

H. Faster CNNs

- R-CNN : It uses a selective-search algorithm which extracts 2000 regions from an image and they are called region proposals. As we have to classify 2000 region proposals per image while training to network it will take a huge amount of time. Thus, it can not be implemented as it takes approx 47 seconds for each test image.
- Fast R-CNN : It uses a similar approach as the RCNN. But, here we feed the total input image to CNN to generate convolutional feature maps instead of feeding the regional proposals. The Fast R-CNN is faster than RCNN in case of training and testing as it does not feed 2000 regional proposals to CNN per image. But Including regional proposals decrease the efficiency of Fast R-CNN during testing time.
- Faster R-CNN: It allows the network learn region proposals. It feeds input images to CNN which provide convolutional Feature maps. Faster R-CNN uses separate networks to predict regional proposals instead of using selective-search algorithms. From the below graph, we can see that Faster R-CNN is much faster than its predecessors.

III. ARCHITECTURE REVIEW OF CNN

CNN is a deep neural network used in visual image processing. CNNs have managed to achieve superhuman performance on some complex visual tasks. They power image search services, self-driving cars, automatic video classification systems, and more. Moreover, CNNs are not restricted to visual perception: they are also successful at other tasks, such as voice recognition or natural language processing (NLP); however, we will focus on visual applications of image recognition.

A. Convolutional Layer

The convolutional layer of a convolutional network is the most important part, which does most of the computational heavy lifting. The convolutional layer's parameters consist of

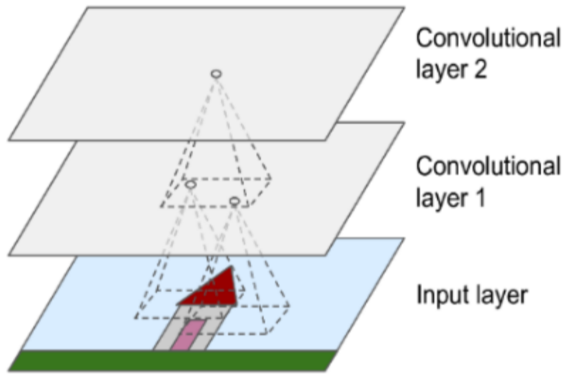


Fig. 1: Convolutional Layer

a set of learnable filters. Every filter is small spatially(width and height), but extends through the full input volume. We slide the filter across width and height of the input volume and compute dot product between filter indices and corresponding input volume locations. as we slide the filter over the volume, we get a 2-d feature map. The network will learn filters that activate when they see some type of visual feature such as an edge of some orientation or a blotch of some color on the first layer. We stack the different feature maps obtained to get the output volume.

B. Pooling Layer

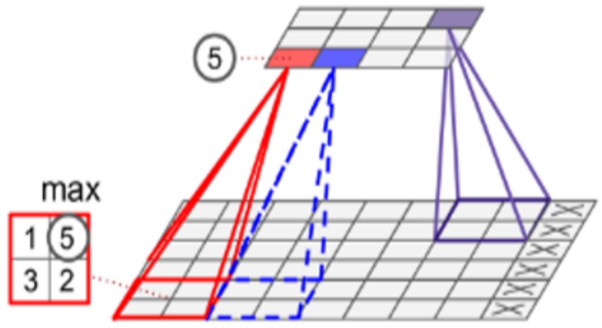


Fig. 2: Pooling Layer

It is common to periodically insert a Pooling layer inbetween successive Conv layers in a ConvNet architecture. Its function is to progressively reduce the spatial size of the representation to reduce the amount of parameters and computation in the network, and hence to also control overfitting. The Pooling Layer operates independently on every depth slice of the input and resizes it spatially, using the MAX operation. The most common form is a pooling layer with filters of size 2x2 applied with a stride of 2 downsamples every depth slice in the input by 2 along both width and height, discarding 75% of the activations. Every MAX operation would in this case be taking a max over 4 numbers (little 2x2 region in some depth slice).

C. Fully-connected layer

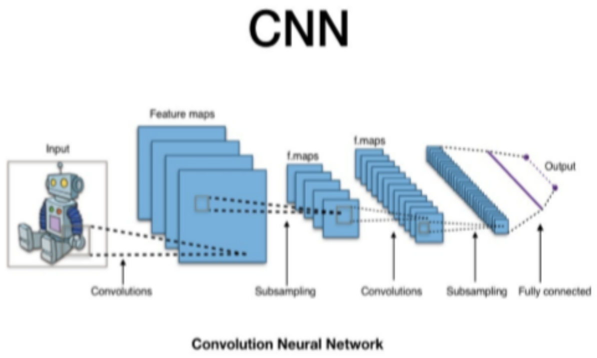


Fig. 3: Network Layers

Neurons in a fully connected layer have full connections to all activations in the previous layer, as seen in regular Neural Networks. Their activations can hence be computed with a matrix multiplication followed by a bias offset.

IV. DATASET

A. Accident Detection and Prediction(CADP) dataset

The dataset consists of 1,416 video segments collected from YouTube, with 205 video segments having full spatio-temporal annotations.This dataset is largest in terms of number of traffic accidents, compared to other contemporary datasets. CADP contains videos collected from YouTube which are captured under various camera types and qualities, weather conditions and edited/resampled videos.

B. DETRAC dataset

The UA-DETRAC benchmark dataset consists of 100 challenging videos captured from real-world traffic scenes (over 140,000 frames with rich annotations, including illumination, vehicle type, occlusion, truncation ratio, and vehicle bounding boxes) for multi-object detection and tracking.

V. TRAINING METHODOLOGIES

We divide the training phase into a number of sub-phases, wherein the learning process is carried sequentilly and different sub-phases are all dependent on the previous instances. We take a brief look into the proposed methodology.

A. DATASET PREPARATION

We use two datasets, namely CADP and DETRAC. CADP has accident images from various CCTV camera footage on youtube all accumulated at an single location. DETRAC, on the other hand, comprises non-accident, normal images that are derived from traffic cameras. We use 28000 accident images from CADP and 23000 non-accident images from DETRAC dataset. The images are divided into directories 'Accident' and 'Non-Accident', and each image is subsequently scaled to 128 $\times$ 128 and then converted to grayscale. The scaled vector with the image matrix and label are appended into a numpy array. After loading the complete set of images, we shuffle the contents of the array(shuffling inserts images with two different labels at random locations, which bring a randomness into the data). The image matrices and labels are store into two separate files on the disk.

Initially, the Dataset was generated from videos that consisted only a small fraction of positive images and large number of false positives. We call this Dataset a ‘Crude Dataset’. This ‘Crude Dataset’ may produce poor validation results and generalization. The efficiency on the ’Crude Dataset’ was mediocre.

In order to overcome the false positives, We manually validated the positive and false positive images, thereby removing most of the false positives. The Dataset was now reduced to 4000 Accident images. We took twice the number of ‘Non-accident’ Images, obtaining a sample size of 12000 images with relative ratio of 1:2 between ‘Accident’ and‘Non-Accident’ images.

B. LOADING TRAINING DATA

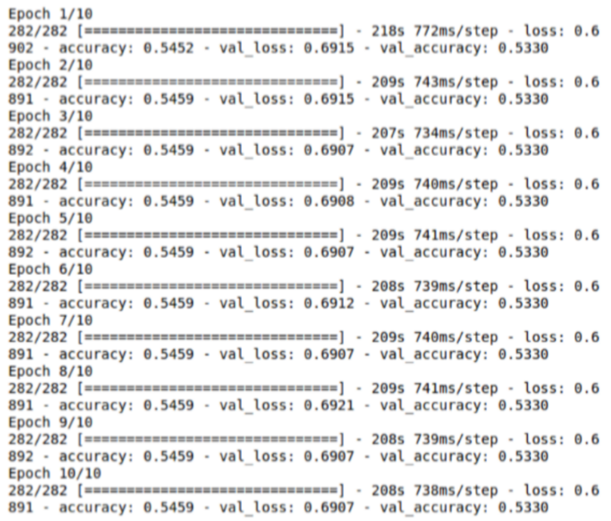
The previously loaded image matrices and labels are retrieved for the learning procdure. We load the previously stored data from the disk. For each image in the loaded set, we normalize the intesity values at spatial locations. Normalization removes bias from dataset by downsizing the large contribution from some features and upsizing small contribution from some features.

C. NETWORK TRAINING

We have sequentially stacks all the layers - keras Conv2D, MaxPooling2D, Activation, Flatten, Dense, Dropout layer. The convolutional layer extracts the feature map from image volume. Activation layer adds the activation function for adding non-linearities. Flatten layer converts all output layers to 1D vectors. ‘ReLu’ and ‘Sigmoid’ are used as activation functions, and ’binary crossentropy’ as our loss function and ’Adam’ optimizer and for accuracy matrix calculation. We take three instances - 1000, 5000 and 10000 images for training the model on ‘Crude Dataset’. We then evaluate the model on the newly created dataset of 12000 images. The whole sample was used for training.

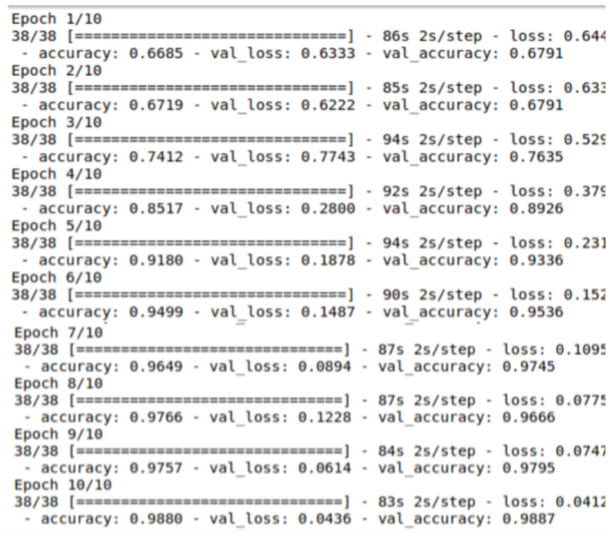
D. TESTING

For Testing, from the three above-mentioned instances, we take 10% instances as the validation set. We observe that for low-size training samples on ‘Crude Dataset’, the accuracy is low. On increasing, the number of instances, the accuracy somewhat increases, but, on further addition to the sample size, the accuracy hardly increases.
For the new Dataset of 12000 images, Testing was done on 350 images sample consisting of ‘Accident’ and ‘Non-accident’ images in equal proportions.



```
Epoch 1/10
282/282 [==============================] - 218s 772ms/step - loss: 0.6
902 - accuracy: 0.5452 - val_loss: 0.6915 - val_accuracy: 0.5330
Epoch 2/10
282/282 [==============================] - 209s 743ms/step - loss: 0.6
891 - accuracy: 0.5459 - val_loss: 0.6915 - val_accuracy: 0.5330
Epoch 3/10
282/282 [==============================] - 207s 734ms/step - loss: 0.6
892 - accuracy: 0.5459 - val_loss: 0.6907 - val_accuracy: 0.5330
Epoch 4/10
282/282 [==============================] - 209s 740ms/step - loss: 0.6
891 - accuracy: 0.5459 - val_loss: 0.6908 - val_accuracy: 0.5330
Epoch 5/10
282/282 [==============================] - 209s 741ms/step - loss: 0.6
892 - accuracy: 0.5459 - val_loss: 0.6907 - val_accuracy: 0.5330
Epoch 6/10
282/282 [==============================] - 208s 739ms/step - loss: 0.6
891 - accuracy: 0.5459 - val_loss: 0.6912 - val_accuracy: 0.5330
Epoch 7/10
282/282 [==============================] - 209s 740ms/step - loss: 0.6
891 - accuracy: 0.5459 - val_loss: 0.6907 - val_accuracy: 0.5330
Epoch 8/10
282/282 [==============================] - 209s 741ms/step - loss: 0.6
891 - accuracy: 0.5459 - val_loss: 0.6921 - val_accuracy: 0.5330
Epoch 9/10
282/282 [==============================] - 208s 739ms/step - loss: 0.6
892 - accuracy: 0.5459 - val_loss: 0.6907 - val_accuracy: 0.5330
Epoch 10/10
282/282 [==============================] - 208s 738ms/step - loss: 0.6
891 - accuracy: 0.5459 - val_loss: 0.6907 - val_accuracy: 0.5330
```

Fig. 4: Validation matrix for n = 10000 instances for ‘Crude Dataset’



```
Epoch 1/10
38/38 [==============================] - 86s 2s/step - loss: 0.644
 - accuracy: 0.6685 - val_loss: 0.6333 - val_accuracy: 0.6791
Epoch 2/10
38/38 [==============================] - 85s 2s/step - loss: 0.633
 - accuracy: 0.6719 - val_loss: 0.6222 - val_accuracy: 0.6791
Epoch 3/10
38/38 [==============================] - 94s 2s/step - loss: 0.529
 - accuracy: 0.7412 - val_loss: 0.7743 - val_accuracy: 0.7635
Epoch 4/10
38/38 [==============================] - 92s 2s/step - loss: 0.379
 - accuracy: 0.8517 - val_loss: 0.2800 - val_accuracy: 0.8926
Epoch 5/10
38/38 [==============================] - 94s 2s/step - loss: 0.231
 - accuracy: 0.9180 - val_loss: 0.1878 - val_accuracy: 0.9336
Epoch 6/10
38/38 [==============================] - 90s 2s/step - loss: 0.152
 - accuracy: 0.9499 - val_loss: 0.1487 - val_accuracy: 0.9536
Epoch 7/10
38/38 [==============================] - 87s 2s/step - loss: 0.1095
 - accuracy: 0.9649 - val_loss: 0.0894 - val_accuracy: 0.9745
Epoch 8/10
38/38 [==============================] - 87s 2s/step - loss: 0.0775
 - accuracy: 0.9766 - val_loss: 0.1228 - val_accuracy: 0.9666
Epoch 9/10
38/38 [==============================] - 84s 2s/step - loss: 0.0747
 - accuracy: 0.9757 - val_loss: 0.0614 - val_accuracy: 0.9795
Epoch 10/10
38/38 [==============================] - 83s 2s/step - loss: 0.0412
 - accuracy: 0.9880 - val_loss: 0.0436 - val_accuracy: 0.9887
```

Fig. 5: Validation matrix for ‘Compact Dataset’

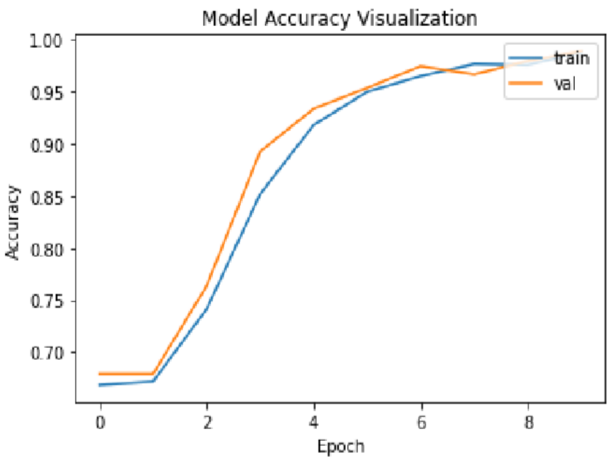


Fig. 6: Validation and Testing Accuracy

VI. PRELIMINARY RESULTS

1. The efficacy obtained so far is moderate. This maybe due to the problem of object detection, the small similarity between accident images, and the lack of more non-linear layers that learns the non-linear features.
2. YOLO(You look only once) Algorithm can be added to classify the various type of objects in the accident and non-accident images. This will enhance the accuracy of the model.
3. Context Augmentation with faster RCNN could further improve the accuracy.
4. Better accuracy can be achieved by optimizing the CADP dataset which conatins false accident images.

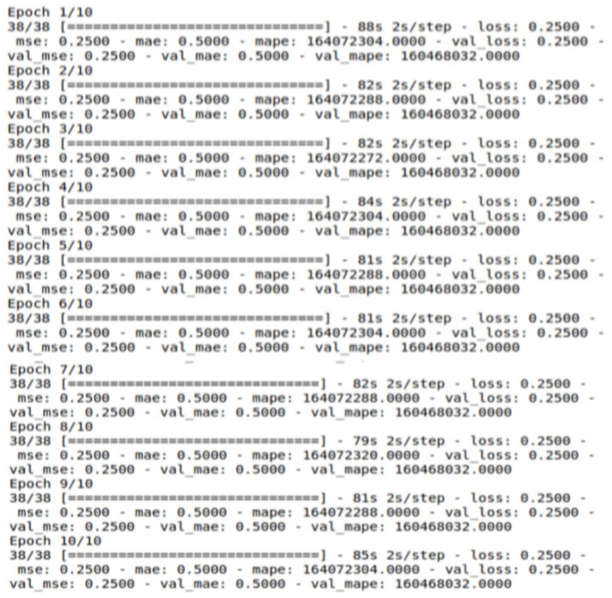
VII. FINAL RESULTS

After manual validation on ‘Crude Dataset’, we obtain a new ‘Compact Dataset’. The result on this ‘Comapct Dataset’ are provided below:

1. The Training and Validation Efficiency obtained was 97% and Testing Accuracy was 68% respectively.
2. The Recall and Precision were 1 and 0.68 respectively.
3. Cosine Similarity was recorded as 95%.

The various error and loss functions are included in the table given below:

Error And Loss Functions			
	Training	Validation	Testing
MSE	0.25	0.25	0.25
MAE	0.5	0.5	0.5
MAPE	164072304.00	160468032.00	2042376.00
Precision	0.67	0.68	0.59
Recall	0.99	1.00	1.00
Cosine S.	0.95	0.96	0.83



```
Epoch 1/10
38/38 [==============================] - 88s 2s/step - loss: 0.2500 -
 mse: 0.2500 - mae: 0.5000 - mape: 164072304.0000 - val_loss: 0.2500 -
val_mse: 0.2500 - val_mae: 0.5000 - val_mape: 160468032.0000
Epoch 2/10
38/38 [==============================] - 82s 2s/step - loss: 0.2500 -
 mse: 0.2500 - mae: 0.5000 - mape: 164072288.0000 - val_loss: 0.2500 -
val_mse: 0.2500 - val_mae: 0.5000 - val_mape: 160468032.0000
Epoch 3/10
38/38 [==============================] - 82s 2s/step - loss: 0.2500 -
 mse: 0.2500 - mae: 0.5000 - mape: 164072272.0000 - val_loss: 0.2500 -
val_mse: 0.2500 - val_mae: 0.5000 - val_mape: 160468032.0000
Epoch 4/10
38/38 [==============================] - 84s 2s/step - loss: 0.2500 -
 mse: 0.2500 - mae: 0.5000 - mape: 164072304.0000 - val_loss: 0.2500 -
val_mse: 0.2500 - val_mae: 0.5000 - val_mape: 160468032.0000
Epoch 5/10
38/38 [==============================] - 81s 2s/step - loss: 0.2500 -
 mse: 0.2500 - mae: 0.5000 - mape: 164072288.0000 - val_loss: 0.2500 -
val_mse: 0.2500 - val_mae: 0.5000 - val_mape: 160468032.0000
Epoch 6/10
38/38 [==============================] - 81s 2s/step - loss: 0.2500 -
 mse: 0.2500 - mae: 0.5000 - mape: 164072304.0000 - val_loss: 0.2500 -
val_mse: 0.2500 - val_mae: 0.5000 - val_mape: 160468032.0000

Epoch 7/10
38/38 [==============================] - 82s 2s/step - loss: 0.2500 -
 mse: 0.2500 - mae: 0.5000 - mape: 164072288.0000 - val_loss: 0.2500 -
val_mse: 0.2500 - val_mae: 0.5000 - val_mape: 160468032.0000
Epoch 8/10
38/38 [==============================] - 79s 2s/step - loss: 0.2500 -
 mse: 0.2500 - mae: 0.5000 - mape: 164072320.0000 - val_loss: 0.2500 -
val_mse: 0.2500 - val_mae: 0.5000 - val_mape: 160468032.0000
Epoch 9/10
38/38 [==============================] - 81s 2s/step - loss: 0.2500 -
 mse: 0.2500 - mae: 0.5000 - mape: 164072288.0000 - val_loss: 0.2500 -
val_mse: 0.2500 - val_mae: 0.5000 - val_mape: 160468032.0000
Epoch 10/10
38/38 [==============================] - 85s 2s/step - loss: 0.2500 -
 mse: 0.2500 - mae: 0.5000 - mape: 164072304.0000 - val_loss: 0.2500 -
val_mse: 0.2500 - val_mae: 0.5000 - val_mape: 160468032.0000
```

Fig. 7: Loss Function and Error Measure

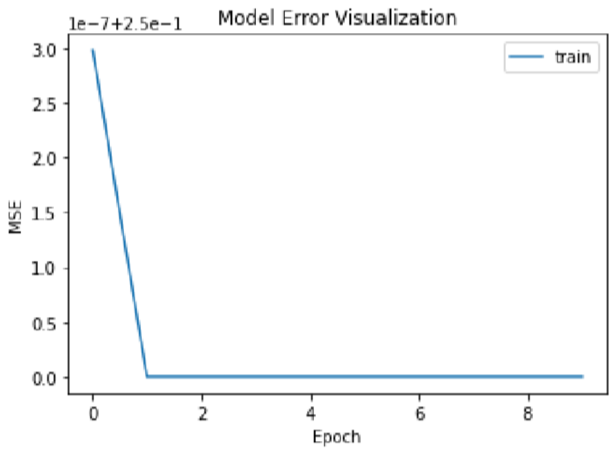


Fig. 8: Mean Squared Error

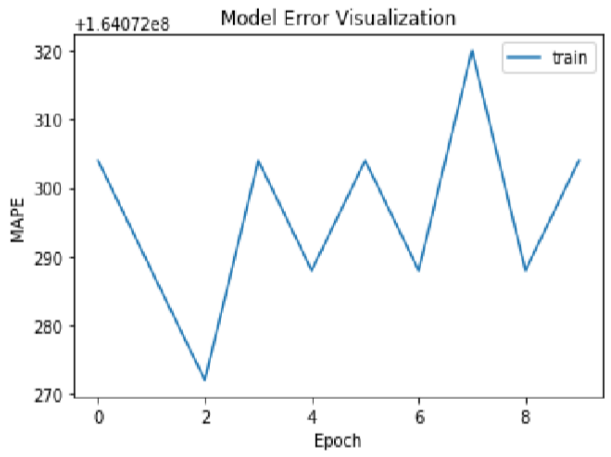


Fig. 9: Mean Absolute Percentage Error

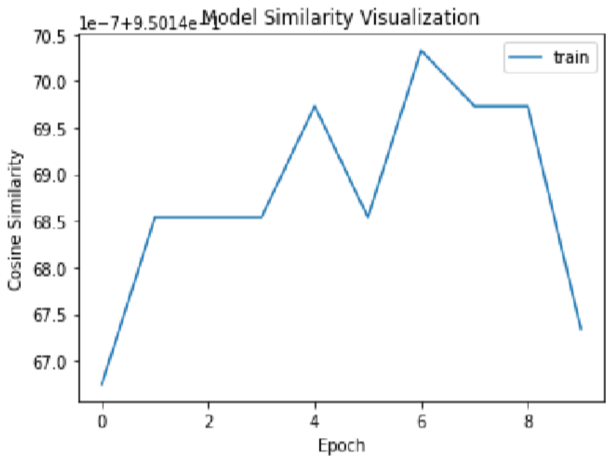
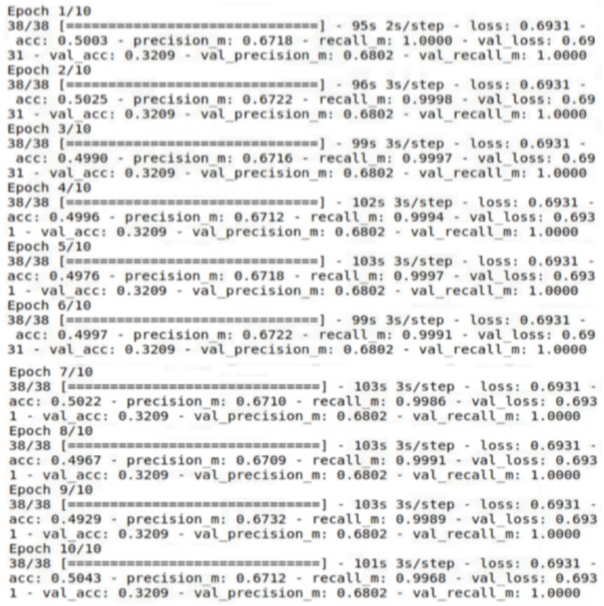


Fig. 10: Cosine Similarity



```
Epoch 1/10
38/38 [==============================] - 95s 2s/step - loss: 0.6931 -
 acc: 0.5003 - precision_m: 0.6718 - recall_m: 1.0000 - val_loss: 0.69
31 - val_acc: 0.3209 - val_precision_m: 0.6802 - val_recall_m: 1.0000
Epoch 2/10
38/38 [==============================] - 96s 3s/step - loss: 0.6931 -
 acc: 0.5025 - precision_m: 0.6722 - recall_m: 0.9998 - val_loss: 0.69
31 - val_acc: 0.3209 - val_precision_m: 0.6802 - val_recall_m: 1.0000
Epoch 3/10
38/38 [==============================] - 99s 3s/step - loss: 0.6931 -
 acc: 0.4990 - precision_m: 0.6716 - recall_m: 0.9997 - val_loss: 0.69
31 - val_acc: 0.3209 - val_precision_m: 0.6802 - val_recall_m: 1.0000
Epoch 4/10
38/38 [==============================] - 102s 3s/step - loss: 0.6931 -
acc: 0.4996 - precision_m: 0.6712 - recall_m: 0.9994 - val_loss: 0.693
1 - val_acc: 0.3209 - val_precision_m: 0.6802 - val_recall_m: 1.0000
Epoch 5/10
38/38 [==============================] - 103s 3s/step - loss: 0.6931 -
acc: 0.4976 - precision_m: 0.6718 - recall_m: 0.9997 - val_loss: 0.693
1 - val_acc: 0.3209 - val_precision_m: 0.6802 - val_recall_m: 1.0000
Epoch 6/10
38/38 [==============================] - 99s 3s/step - loss: 0.6931 -
 acc: 0.4997 - precision_m: 0.6722 - recall_m: 0.9991 - val_loss: 0.69
31 - val_acc: 0.3209 - val_precision_m: 0.6802 - val_recall_m: 1.0000

Epoch 7/10
38/38 [==============================] - 103s 3s/step - loss: 0.6931 -
acc: 0.5022 - precision_m: 0.6710 - recall_m: 0.9986 - val_loss: 0.693
1 - val_acc: 0.3209 - val_precision_m: 0.6802 - val_recall_m: 1.0000
Epoch 8/10
38/38 [==============================] - 103s 3s/step - loss: 0.6931 -
acc: 0.4967 - precision_m: 0.6709 - recall_m: 0.9991 - val_loss: 0.693
1 - val_acc: 0.3209 - val_precision_m: 0.6802 - val_recall_m: 1.0000
Epoch 9/10
38/38 [==============================] - 103s 3s/step - loss: 0.6931 -
acc: 0.4929 - precision_m: 0.6732 - recall_m: 0.9989 - val_loss: 0.693
1 - val_acc: 0.3209 - val_precision_m: 0.6802 - val_recall_m: 1.0000
Epoch 10/10
38/38 [==============================] - 101s 3s/step - loss: 0.6931 -
acc: 0.5043 - precision_m: 0.6712 - recall_m: 0.9968 - val_loss: 0.693
1 - val_acc: 0.3209 - val_precision_m: 0.6802 - val_recall_m: 1.0000
```

Fig. 11: Precision and Recall

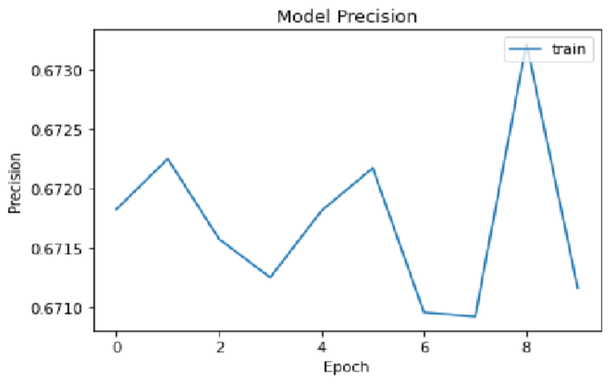


Fig. 12: Precision

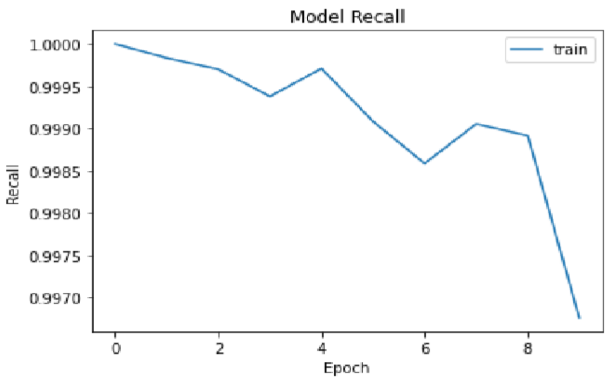


Fig. 13: Recall

VIII. Future Scope

Integration with real time APIs can be done to compute probabilistic results based on traffic data. Integration with traffic monitoring can also be done to predict expected jams and diversions. Dataset can further be improved by adding more images.

IX. Conclusion

The results obtained so far is impressive. The Model Accuracy is poised at 68% during Testing. This accuracy can be increased by training the model on larger sample. Due to the manual overhead of obtaining positive 'Accident' images, obtainig an overwhelming number was not possible. But, if a larger dataset is available, the test accuracy can further be enhanced. In Real-Time system, feedback from traffic-sensing models and APIs could further provide a validation for model prediction. If the model is combined with above-mentioned APIs, then a better Real-Time prediction can be obtained. One such example is the 'Tom Tom' APIs available that provide location-based traffic sensing. We further look forward to accomodate this feature into the model.

References

[1] Li, Zewen, Wenjie Yang, Shouheng Peng, and Fan Liu. "A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects." arXiv preprint arXiv:2004.02806 (2020).

[2] Shah, Ankit, Jean Baptiste Lamare, Tuan Nguyen Anh, and Alexander Hauptmann. "Accident forecasting in CCTV traffic camera videos." arXiv preprint arXiv:1809.05782 (2018).

[3] Wang, Chen, Yulu Dai, Wei Zhou, and Yifei Geng. "A vision-based video crash detection framework for mixed traffic flow environment considering low-visibility condition." Journal of advanced transportation 2020 (2020).

[4] Dong Yin, Fan Zhang, Kun Wang (2012), "A Method for Traffic Collision Detection", Applied Mechanics and Materials Vols. 220-223 (2012) pp 2606-2610.

[5] S. Kamijo, Y. Matsushita, K. Ikeuchi, and M. Sakauchi, "Traffic monitoring and accident detection at intersections," IEEE Transactions on Intelligent Transportation Systems, vol. 1, no. 2, pp. 108–118, 2000.

[6] E. Paul, "Computer vision-based accident detection in traffic surveillance," in Proceedings of the 2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT), Kanpur, India, July 2019.

[7] Akshit Diwan, Vandit Gupta1, Chaitanya Chadha, "Accident Detection Using Mask R-CNN ", International Journal for Modern Trends in Science and Technology, 7(01): 69-72, 2021, ISSN: 2455-3778.

[8] Zhenbo Lu, Wei Zhou, Shixiang Zhang, and Chen Wang, "A New Video-Based Crash DetectionMethod: Balancing Speed and Accuracy Using a Feature Fusion Deep Learning Framework", Journal of Advanced Transportation Volume 2020, Article ID 8848874, 12 pages.

[9] Dhananjai Chand , Savyasachi Gupta , Ilaiah Kavati. "Computer Vision based Accident Detection for Autonomous Vehicles." Department of Computer Science and Engineering National Institute of Technology Warangal, India-506004.

[10] Hoai Nam Vu, Ngoc Hung Dang, "An Improvement of Traffic Incident Recognition by Deep Convolutional Neural Network." International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN: 2278-3075, Volume-8 Issue-1, November 2018.