International Journal of Engineering Applied Sciences and Technology, 2021
Vol. 6, Issue 5, ISSN No. 2455-2143, Pages 109-114
Published Online September 2021 in IJEAST (http://www.ijeast.com)

COMPUTER VISION BASED TRAFFIC RULE BREACH DETECTION

Sunim Acharya, Sujan Poudel, Shreeya Dangol, Saragam Subedi
Department of Computer Engineering, Kathmandu Engineering College
Tribhuvan University

***Abstract*— This paper is about the detection of traffic rule breach via computer vision which takes the feed from the traffic surveillance system, processes the video feed, detects the breach and alerts the traffic police. The number of traffic accidents is on the rise with the increasing number of vehicles. Traffic breach is the biggest cause of accidents. So, to mitigate this problem our system processes the CCTV camera feed in real-time, detects the traffic rule breach events and sends the push notification to the android based application of the traffic police stationed nearby; so, further actions can be taken. As this system detects breach faster than humans, the concerned authoritarian department will be at ease in implementing safe roads accurately. This system acts as an add-on to the current video surveillance system rather than building new infrastructure. Thus, the output of this system can be used not only or safety and security purposes but as well as for analytical purposes with effective traffic monitoring at a lower cost. Hence, this system aids law enforcement agencies in implementing road safety efficiently and effectively ensuring smooth traffic flow.**

***Keywords*— detection; CCTV; traffic breach; surveillance; push notification;**

I. INTRODUCTION

Traffic on roads consists of road users including pedestrians, ridden or herded animals, vehicles and other conveyances, either singly or together, while using the public way for purposes of travel. Traffic laws govern traffic and regulate vehicles, while rules of the road are both the laws and the informal rules that may have developed over time to facilitate the orderly and timely flow of traffic.
Nowadays, we are using CCTV cameras to capture the video feed of traffic to identify the traffic rule breach events by personnel in a control room or to use the footage as evidence against a breach. They are video cameras, capturing and saving the video feed to temporary or permanent storage media waiting for a human to analyze. The goal of our system is to automate the traffic rules breach detection system and make the task of surveillance easier for the traffic police department to monitor the traffic and take action against the violated vehicle owner in a fast and efficient way.

II. LITERATURE SURVEY

A. Vision-based real-time traffic accident detection, Zuhui (2014)

With the growth of the number of vehicles, the number of traffic accidents is rapidly rising. Therefore, taking this thing in consideration Xiaoling Wang, Li-Min Meng, Biaobiao Zhang, and K.-L.Du made a traffic rule violation detection system using movement detection and tracking[1]. They used the background difference method and the inter-frame difference method for movement detection. The dynamical background update method based on the wavelet transform, which was combined difference and feature-based tracking to detect moving vehicles.

B. A Video-based Traffic Violation Detection System(2013)

This system can detect traffic violations, such as running red lights, speeding, and vehicle regress in real-time. This system proposed an improved background-updating algorithm by using the wavelet transform on the dynamic background and then track moving vehicles by feature-based tracking method. A complete traffic violation detection system is realized in C++ with OpenCV. The system combines the background difference method and feature-based tracking method that enables accurate detection and tracking[2].

C. Traffic Rules Violation Detection System Using Computer Vision

The goal of the project was to automate the traffic rules violation detection system and make it easy for the traffic police department to monitor the traffic and take action against the violated vehicle owner in a fast and efficient way. Detecting and tracking the vehicle and their activities accurately was the main priority of the system. The system could detect three major violations: Signal, Parking and Direction Violation[3].

III. FEATURES

A. When a vehicle crosses a predefined line under Red Light, the vehicle has breached a rule.
B. When a vehicle is moving in the opposite direction of the lane, the vehicle has breached a rule.
C. When a vehicle is immobile for a specific duration in no parking zone, the vehicle has breached a rule.

IV. IMPLEMENTATION

A. *Image Processing*

1) *Grayscaling and Blurring*

Gaussian Blur method is used to grayscale and blur the image.

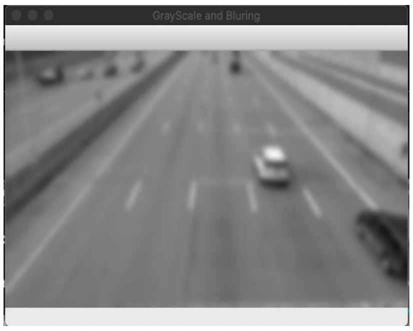


Figure. 1 Grayscaling and blurring of video feed

2) *Background Subtraction*

Object Area is based on the subtraction between the current frame and the previous frame as a reference[4].

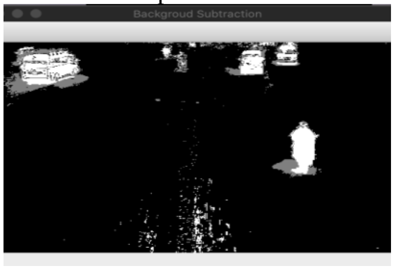


Figure. 2 Background Subtraction

$dst(I) = saturate(|scr1(I) - scr2(I)|)$

Where scr1 = current frame; scr2 = reference frame; dst = subtracted frame

3) *Binary Threshold*

For accuracy, Binarization method removes all the holes and noises from the frame[5].

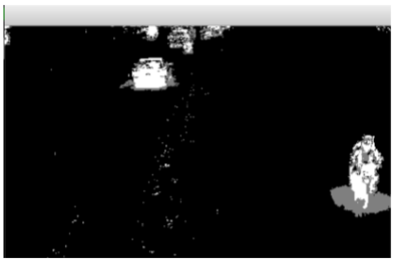


Figure. 3 Binary Threshold

$dst(x, y) = maxval, \text{ if } scr(x, y) > thresh, 0 \text{ otherwise}$

4) *Dilation and finding contour*

From the thresholded image, dilation is done along with finding contour whose area is used to compare with a predefined size and then feed into a neural network if it is within the threshold range[6].

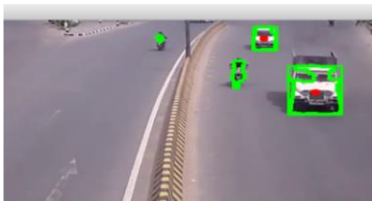


Figure. 4 Dilation and Finding Contour

B. *Classification*

The image after preprocessing is then fed into a custom trained neural network to identify two classes: **Car** and **Bike**.

A deep neural network (DNN) is an artificial neural network (ANN) with multiple hidden layers between the input and output layers DNNs can model complex non-linear relationships. DNN architectures generate compositional models where the object is expressed as a layered composition of primitives[7]. The extra layers enable the composition of features from lower layers, potentially modeling complex data with fewer units than a similarly performing shallow network. The overview of the structure of Neural network is:

1) *Preprocess and load data*

We process the data before feeding to the neural network and also visualize data which will gain insight into the data.

2) *Define model*

We specify the number of hidden layers in the neural network and their size, the input and output size.

3) *Loss and optimizer*

We define the loss function according to our task along with optimizer to use with the learning rate and other hyperparameters of the optimizer.

4) *Fit model*

This is the training step of the neural network where we define the number of epochs for which we need to train the neural network.

More formally, call *aji* the activation (output) of the j^{th} neuron in the i^{th} layer, where a 1 j is the j^{th} element in the input vector[8]. Then we can relate the next layer's input to its previous via the following relation:

$aj\ i = \sigma(\sum(wjk\ i \cdot ak\ i\text{-}1) + bj\ i)\ k$

Where σ is the activation function; $w$jk i is the weight from the k^{th} neuron in the $(i\text{-}1)^{th}$ layer to the j^{th} neuron in the i^{th} layer; bj i is the bias of the j^{th} neuron in the i^{th} layer; $a$ji represents the activation value of the j^{th} neuron in the i^{th} layer.

The fully connected layer was trained using Keras with 1424 training set and 286 test set by scraping from the internet. The major steps are:

1) Initial Sequential CNN

The role of the ConvNet is to reduce the images into a form that is easier to process, without losing features to get a good prediction.

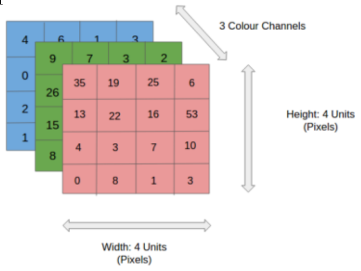


Figure. 5 4x4x3 RGB Image

2) Two Convolution Layers

The filter moves to the right with a specific Stride Value till it parses the complete width. Moving on, it hops down to the beginning (left) of the image with the same Stride Value and repeats the process until the entire image is traversed[9].

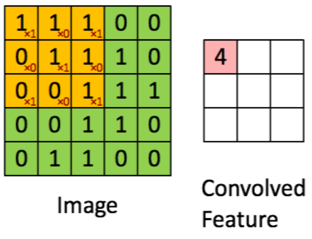


Figure. 6 Convoluting 5x5x1 image with 3x3x1 kernel to get a 3x3x1 feature

3) Max Pooling

Max Pooling returns the maximum value from the portion of the image covered by the Kernel to decrease the computational power required to process the data through dimensionality reduction.

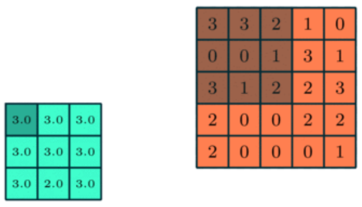


Figure. 7 3x3 pooling over 5x5 convolved feature

4) Flattening

As we have converted the input images into a suitable form, we flatten the image into a column vector. The flattened output is fed to a feed-forward neural network and backpropagation applied to every iteration of training [10]. Over a series of epochs, the model is able to distinguish between dominating and specific low-level features in images and classify them using the Softmax Classification technique.

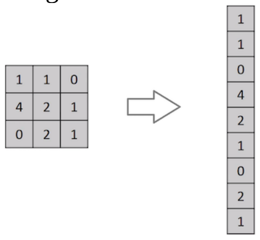
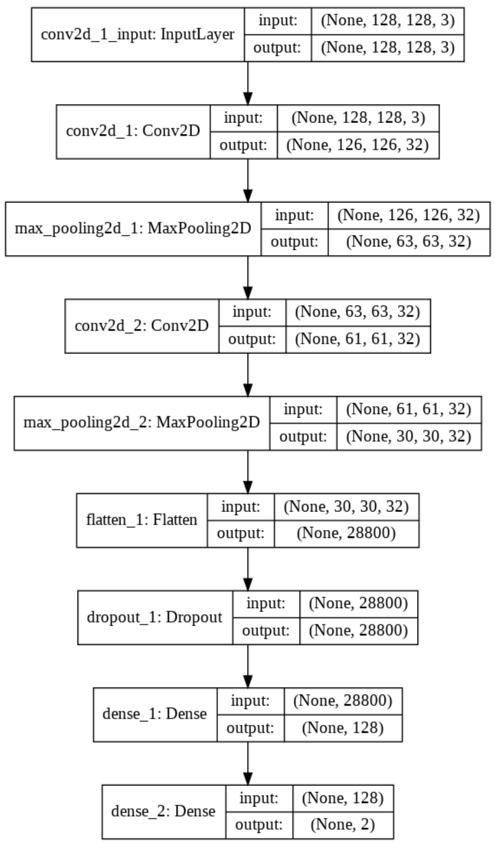


Figure. 8 Flattening of a 3x3 image matrix into a 9x1 vector



conv2d_1_input: InputLayer — input: (None, 128, 128, 3) — output: (None, 128, 128, 3)
conv2d_1: Conv2D — input: (None, 128, 128, 3) — output: (None, 126, 126, 32)
max_pooling2d_1: MaxPooling2D — input: (None, 126, 126, 32) — output: (None, 63, 63, 32)
conv2d_2: Conv2D — input: (None, 63, 63, 32) — output: (None, 61, 61, 32)
max_pooling2d_2: MaxPooling2D — input: (None, 61, 61, 32) — output: (None, 30, 30, 32)
flatten_1: Flatten — input: (None, 30, 30, 32) — output: (None, 28800)
dropout_1: Dropout — input: (None, 28800) — output: (None, 28800)
dense_1: Dense — input: (None, 28800) — output: (None, 128)
dense_2: Dense — input: (None, 128) — output: (None, 2)

Figure. 9 Model plot of the Fully Connected Layer

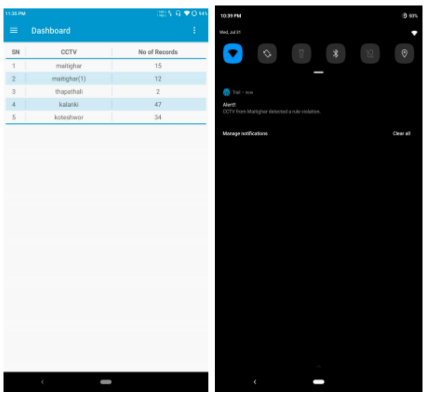
The classifier was saved in a .h5 extension which is then used by the pre-processed image to identify if the fed image falls under either of the two classes. If the image is identified, it is sent to a Django-server.

C. Server Deployment

If a breach is detected, the image is sent to a Django Server deployed on Heroku which saves the image in Django Model and sends push notification to the authenticated user ie. Traffic Police.

D. Push Notification

After inserting the image into the database with timestamp and photo, a push notification is sent to nearby stationed traffic police via Firebase Cloud Messaging (FCM) for further action.



SN	CCTV	No of Records
1	maitighar	15
2	maitighar(1)	12
3	thapathali	2
4	kalanki	47
5	koteshwor	34

Alert!
CCTV from Maitighar detected a rule violation.

Figure. 10 Screenshot of the Mobile application and FCM notification

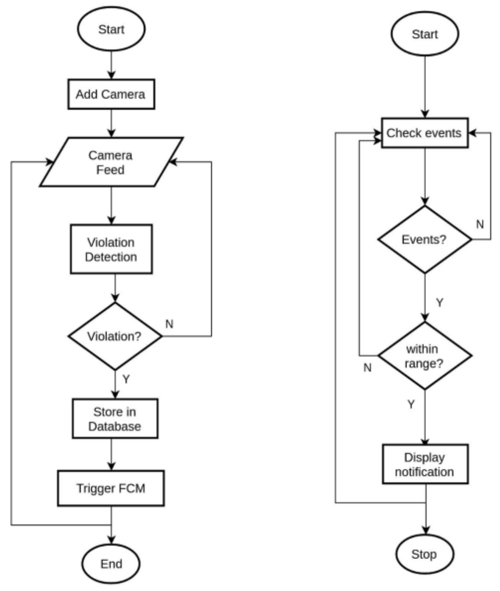


Figure. 11 Flow Chart of the system

V. RESULTS

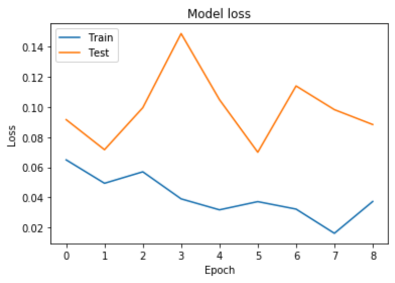
A. Classification Training Result

To evaluate the achieved results, we employ statistical metrics: accuracy and loss.

TABLE I
TRIAL 1 IN INTEL 6100

Epoch	Loss	Accuracy	Validation Loss	Validation Accuracy
1	1.1818	0.7817	0.1358	0.9508
2	0.1459	0.9394	0.1059	0.9611
3	0.1689	0.9331	0.1270	0.9462
4	0.0725	0.9725	0.0789	0.9663
5	0.0980	0.9634	0.1528	0.9456

Where epoch = 5; steps per epoch = 71



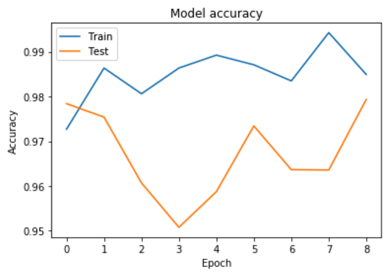


Figure. 12 Model Loss vs Model Accuracy (Trial I)

TABLE II
TRIAL 1I IN GTX 1070

Epoch	Loss	Accuracy	Validation Loss	Validation Accuracy
1	0.0245	0.9923	0.2638	0.9301
2	0.0297	0.9894	0.1549	0.9741
3	0.0795	0.9704	0.2142	0.9005
4	0.0633	0.9796	0.1298	0.9611
5	0.0368	0.9852	0.0829	0.9741
6	0.0357	0.9845	0.0800	0.9731
7	0.0410	0.9866	0.0878	0.9611
8	0.0235	0.9915	0.1028	0.9741

9	0.0138	0.9958	0.1892	0.9489

Where epoch = 9; steps per epoch = 44

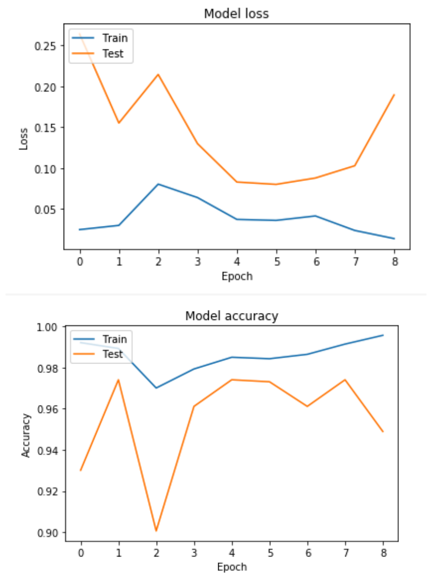


Figure. 13 Model Loss vs Model Accuracy (Trial II)

B. *Testing Devices*

TABLE III
DEVICES USED TO TEST

Device	RAM	CPU	GPU	Runtime	Compute	Processor
Macbook Pro 2015	8 GB	i5	Intel 6100	Common	1x-4x	2.7 Ghz
Dell Inspiron 15 5577	8GB	i5	GTX 1050	Turbo Boost	64 bit	2.5 GHz

C. *Heroku Server Configuration*

TABLE IV
CONFIGURATION OF HEROKU SERVER

Dyno type	RAM	CPU (Shared)	Dedicated	Runtime	Compute	Dyno consumed
standard-1x	512 MB	1x	no	Common	1x-4x	1

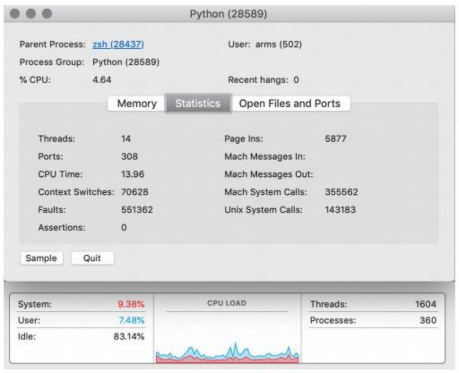
D. *Recommended system requirement for control room*

TABLE IV
CONFIGURATION OF HEROKU SERVER

CPU/GPU	GPUs	vCPUs	System Memory (GB)	Max number of persistent disks (PDs)	Max Total PD size (TB)
Intel(R) Xeon(R) CPU @ 2.5 GHz	GTX - 1050 +	2 +	4 +	8 +	30 +

VI. RESULTS

E. *Average Performance vs Stress*



Python (28589)

Parent Process: zsh (28437)   User: arms (502)
Process Group: Python (28589)
% CPU: 4.64   Recent hangs: 0

Memory | Statistics | Open Files and Ports

Threads:	14	Page Ins:	5877
Ports:	308	Mach Messages In:	
CPU Time:	13.96	Mach Messages Out:	
Context Switches:	70628	Mach System Calls:	355562
Faults:	551362	Unix System Calls:	143183
Assertions:	0		

Sample   Quit

System:	9.38%	Threads:	1604
User:	7.48%	Processes:	360
Idle:	83.14%		

CPU LOAD

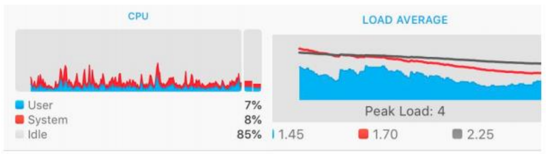


Figure. 14 CPU Stress vs Load Average

Figure. 14 shows the average stress the system has over a specific period of time, at an average of 7%, when using the system to detect traffic rule breach. The performance is bound to change based on the device used to test.

VII. CONCLUSION

To conclude, Trail is a traffic rule violation detection system that processes the CCTV camera feed in real-time, detects the traffic rule violation events and sends the push notification to the android based application to take further actions. This system detects violations faster than humans, the concerned authoritarian department will be at ease in implementing safe roads accurately. This system acts as an add-on to the current traffic surveillance system. However, the system has its own set of drawbacks when having to detect multiple vehicles at the same time or there is a minute occurrence of the breach which might go undetected.

The goal of the system is to automate the traffic rules violation detection system and make the task of surveillance easier for the traffic police department.

VIII. ACKNOWLEDGEMENT

This research was supported by the Department of Computer Engineering, Kathmandu Engineering College, Tribhuvan

University. The basic processing step was based on *Traffic Rules Violation Detection System Using Computer Vision*, significantly the PyQT User-Interface, while the CCTV recording was provided by Bikash Thapaliya, Kantipur Publication extracted from 5wh Channel. The authors appreciate their data support.

IX. REFERENCES

[1] Wang, Xiaoling & Meng, Li-Min & Zhang, Biaobiao & Lu, Junjie & Du, K.-L. (2013). A video-based traffic violation detection system. Proceedings - 2013 International Conference on Mechatronic Sciences, Electric Engineering and Computer, MEC 2013. 1191-1194. 10.1109/MEC.2013.6885246.

[2] A video-based traffic violation detection system, 14 July, <URL:https://www.researchgate.net/publication/271545712_A_video-based_traffic_violation_detection_system>

[3] Traffic Rules Violation DetectionSystem Using Computer Vision, 20 July,<URL:https://drive.google.com/file/d/15PrE6yta_nEEbEySDklNR5TIa3SGkgTV/view>

[4] N. Singla, "Motion Detection Based on Frame Difference Method," in International Journal of Information & Computation Technology. ISSN 0974-2239 Volume 4, Number 15 (2014), pp. 1559-1565

[5] Guillaume Lazzara, Thierry Géraud. Efficient Multiscale Sauvola's Binarization. International Journal on Document Analysis and Recognition, Springer Verlag, 2014, 17 (2), pp.105-123. ff10.1007/s10032- 013-0209-0ff. ffhal-02181880f

[6] Basic motion detection and tracking with Python and OpenCV, 8 March,<URL:https://www.pyimagesearch.com/2015/05/25/basic-motion-detection-and-tracking-with-python-and-opencv/>

[7] Deep Neural Network- Python Deep Learning Tutorial, 19 April, <URL:https://www.tutorialspoint.com/python_deep_learning/python_deep_learning_deep_neural_networks.htm>

[8] Building our first neural network in Keras, 22 April, <URL:https://towardsdatascience.com/building-our-first-neural-network-in-keras-bdc8abbc17f5>

[9] Albawi, Saad & Abed Mohammed, Tareq & ALZAWI, Saad. (2017). Understanding of a Convolutional Neural Network. 10.1109/ICEngTechnol.2017.8308186.

[10] A Comprehensive Guide to Convolutional Neural Networks, 05 May, <URL:https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53>