Project Report

Indian Institute of Information Technology SriCity

- Ankur Gupta (UG3)
S20170010013

1) Problem Statement

To monitor the traffic density considering the varying and different climatic conditions

1.1) Problem Objectives

- Estimating the traffic density flow from the surveillance camera data.
- Tracking of vehicles
- categorizing the vehicles
- Vehicle count.

1.2) Motivation

- *To reduce accidents in the monitored area*
- *To reduce congestion in the monitored area*
- *Diagnosis of the type and cause of overcrowded conditions*

1.3) Requirements

1.3.1) Packages

absl-py==0.7.1	astor==0.8.0	attrs==19.1.0	backcall==0.1.0
bleach==3.1.0	certifi==2018.8.24	colorama==0.4.1	cycler==0.10.0
cython==0.29.10	decorator==4.4.0	defusedxml==0.6.0	entrypoints==0.3
filterpy==1.4.5	gast==0.2.2	google-pasta==0.1.7	grpcio==1.21.1
h5py==2.9.0	imutils==0.5.2	ipykernel==5.1.1	ipython==7.6.0
ipython-genutils==0.2.0	ipywidgets==7.4.2	jedi==0.14.0	Jinja2==2.10.1
joblib==0.13.2	jsonschema==3.0.1	jupyter==1.0.0	jupyter-client==5.2.4
jupyter-console==6.0.0	jupyter-core==4.5.0	Keras-Applications==1.0.8	Keras-Preprocessing==1.1.0
kiwisolver==1.1.0	lxml==4.3.4	Markdown==3.1.1	MarkupSafe==1.1.1
matplotlib==3.0.3	mistune==0.8.4	nbconvert==5.5.0	nbformat==4.4.0
notebook==5.7.8	numpy==1.16.4	object-detection==0.1	opencv-python==4.1.0.25
pandas==0.24.2	pandocfilters==1.4.2	parso==0.5.0	pickleshare==0.7.5
Pillow==6.0.0	prometheus-client==0.7.1	prompt-toolkit==2.0.9	protobuf==3.8.0
Pygments==2.4.2	pyparsing==2.4.0	pyrsistent==0.15.2	python-dateutil==2.8.0
pytz==2019.1	pywinpty==0.5.5	pyzmq==18.0.2	qtconsole==4.5.1
scikit-learn==0.21.2	scipy==1.3.0	Send2Trash==1.5.0	six==1.12.0
tensorboard==1.14.0	tensorflow-estimator==1.14.0	tensorflow-gpu==1.14.0	termcolor==1.1.0
terminado==0.8.2	testpath==0.4.2	tornado==6.0.3	tqdm==4.32.2

traitlets==4.3.2	wcwidth==0.1.7	webencodings==0.5.1	Werkzeug==0.15.4
widgetsnbextension==3.4.2	win-unicode-console==0.5	wincertstore==0.2	wrapt==1.11.2

1.3.2) Hardware Utilized to achieve the results

- RAM : 16 Gb
- GPU : GTX GeForce 1050Ti - 4Gb
- CPU : Intel Core I7-7700HQ

2)DATASET

2.1) IIT MADRAS (IITM-HeTra)-D2[1]

This dataset was collected by IIT madras for Indian road conditions. It consists of 1417 images having 3294 two-wheelers, 279 heavy motor vehicles (HMV), 2148 cars, and 598 auto-rickshaws.

2.2.1) Dataset Organization

Training Samples	Validation Samples	Classes
1202	215	We have used 4 classes 1) car 2) bus 3) auto 4) person

2.2) Berkley Deep Drive Dataset[2]

BDD100K is a collection of 100K traffic images in different lighting and weather conditions. It was having 12 classes out of which we have removed 3 as per our requirements. Images containing only those classes that were not in the interest have also been removed.

2.2.1) Dataset Organization

Training Samples	Validation Samples	Classes
70000	9865	We have used 9 classes 1) car 2) bus 3) person 4) truck 5) traffic sign 6) traffic light 7) bike 8) rider 9) motor

2.3) Indian Driving Dataset[3]

The dataset consists of images obtained from a front facing camera attached to a car in Indian driving conditions. The images are mostly of 1080p resolution, but there is also some images with 720p and other resolutions.

2.3.1) Dataset Organization

Training Samples	Validation Samples	Classes
31569	10225	We have used 15 classes 1) bicycle 2) bus 3) traffic sign 4) Training Samples 5) motorcycle 6) car 7) traffic light 8) person 9) vehicle fallback 10) truck 11) autorickshaw 12) animal 13) caravan 14) rider 15) trailer

2.4) Indian Driving Dataset Revised[3]

In the revised version of the Idd dataset 6 classes have been removed and images having only those classes have also been removed.

2.4.1) Dataset Organization

Training Samples	Validation Samples	Classes
30254	9849	We have used 9 classes 1) bicycle 2) bus 3) motorcycle 4) car 5) traffic light 6) pedestrian 7) truck 8) autorickshaw 9) rider

3) Object detection

In Object detection we identify the location of objects in an image, classify them based on the categories we train our dataset on.

Figure 1: Object detection

3.1) Models[4]

Model	Speed(ms)	COCO(mAP)
Faster rcnn resnet50	89	30
Faster rcnn resnet101	106	32
Faster rcnn inception v2	58	28
Faster rcnn inception resnet v2 atrous	620	37

4) Results

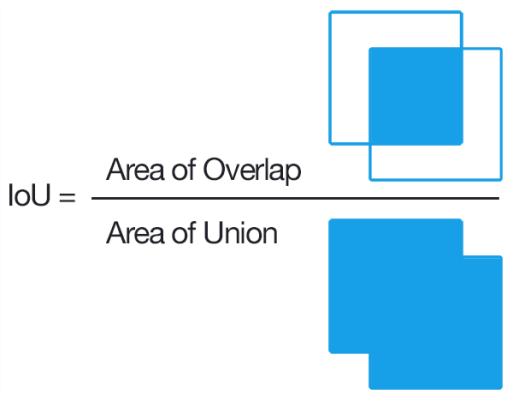
4.1) Metrics

4.1.1) IOU (Intersection Over Union)[5]:

Intersection over Union is an evaluation metric used to measure the accuracy of an object detector on a particular dataset

The *ground-truth bounding boxes* (i.e., the hand labeled bounding boxes from the Validation Sampling set that specify *where* in the image our object is).

The *predicted bounding boxes* from our model.



4.1.2) MAP (Mean Average Precision) [6]:

AP (Average precision) is a popular metric in measuring the accuracy of object detectors like Faster R-CNN. Average precision computes the average precision value for recall value over 0 to 1.

Precision & recall

Precision measures how accurate is your predictions. i.e. the percentage of your predictions are correct.

Recall measures how good you find all the positives.

$$\text{Precision} = \frac{TP}{TP+FP}$$

$$\text{Recall} = \frac{TP}{TP+FN}$$

TP = True Positive TN = True Negative FP = False Positive FN = False Negative

For Mean Average Precision we take the Mean of the Average precision over all Classes

4.2) Tabular Representation

No	Dataset	Split	#	Model	steps	R mAP 0.5	C mAP 0.5
1.	IIT-M	Train-1202 Val-215	04	Faster Rcnn Inception v2	61460	**89.68%**	88.00%
2.	IDD original	Train-31569 Val-10225	15	Resnet 101 coco	70756	21.71%	20.23%
3.	IIT-M	Train- Val-215	04	Faster Rcnn Resnet 101 coco	115895	**90.99%**	89.44%
4.	IDD revised	Train-30254 Val-9849	09	Faster rcnn Resnet 50	89738	34.31%	31.58%
5.	BDD	Train- Val-	09	Faster Rcnn inception v2 Autrous	62623	14.36%	12.25%
6.	IDD revised	Train-30254 Val-9849	09	Faster rcnn Resnet 101	138051	35.49%	33.23%

4.3) Detection Images

Figure 2: Day Time - Anna Rao circle

Figure 3: Night Time – East Police Station

5) Challenges

- Elimination of person shadow and Illumination effect
- Multiple contours detected for the same entity
- Occlusion between vehicles
- Remodifying the threshold part and counting the contours based on the frame
- Classification

6) Conclusion

We have implemented object detection using tensorflow object detection API. We have trained different models on different datasets and observed their results using Mean Average Precision. We have also implemented Centroid Based Tracking to track the position of the vehicles in the frame. We have counted the incoming and outgoing vehicles along a path.

7) References

[1] https://www.kaggle.com/deepak242424/iitmhetra
[2] https://bair.berkeley.edu/blog/2018/05/30/bdd/
[3] https://idd.insaan.iiit.ac.in/dataset/details/
[4] https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/ detection_model_zoo.md
[5] https://www.pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/
[6] https://towardsdatascience.com/breaking-down-mean-average-precision-map-ae462f623a52