Traffic Flow Estimation based on Deep Learning for Emergency Traffic Management using CCTV Images

Nilani Algiriyage*
Joint Centre for Disaster Research, Massey University
r.nilani@massey.ac.nz

Raj Prasanna
Joint Centre for Disaster Research, Massey University
r.prasanna@massey.ac.nz

Emma E H Doyle
Joint Centre for Disaster Research, Massey University
e.e.hudson-doyle@massey.ac.nz

Kristin Stock
Institute of Natural and Mathematical Sciences, Massey University
k.stock@massey.ac.nz

David Johnston
Joint Centre for Disaster Research, Massey University
d.m.johnston@massey.ac.nz

ABSTRACT

Emergency Traffic Management (ETM) is one of the main problems in smart urban cities. This paper focuses on selecting an appropriate object detection model for identifying and counting vehicles from closed-circuit television (CCTV) images and then estimating traffic flow as the first step in a broader project. Therefore, a case is selected at one of the busiest roads in Christchurch, New Zealand. Two experiments were conducted in this research; 1) to evaluate the accuracy and speed of three famous object detection models namely faster R-CNN, mask R-CNN and YOLOv3 for the data set, 2) to estimate the traffic flow by counting the number of vehicles in each of the four classes such as car, bus, truck and motorcycle. A simple Region of Interest (ROI) heuristic algorithm is used to classify vehicle movement direction such as "left-lane" and "right-lane". This paper presents the early results and discusses the next steps.

Keywords

CCTV Big Data, YOLOv3, Traffic Flow Estimation.

INTRODUCTION

Traffic flow estimation is important for urban planning and management of road traffic infrastructure. It is also essential to have a good understanding of the flow of traffic in order to manage emergencies (eg., to re-route traffic through alternative routes) (Yuan et al. 2013). Researchers and developers have recently focused on deep neural networks for traffic prediction, particularly in smart urban cities (Lv et al. 2014; Kang et al. 2017). Emergency Traffic Management (ETM) can be described as a specific case of traffic management requiring extensive planning to ensure secure and effective egress. The causes of traffic emergencies can be small-scale (e.g., vehicle crash) or large-scale (e.g., earthquake or tsunami). They can also be planned (e.g., scheduled maintenance, noticed evacuation before a disaster) or unplanned. Wrong decisions made without a clear picture of the situation have led to multiple unfortunate incidents resulting in dozens of human casualties during mass evacuations (Carpender et al. 2006; Harten et al. 2018).

*corresponding author

Identification of traffic flow is the first step in consolidated planning of managing traffic emergencies. Today, closed-circuit television (CCTV) systems are extremely common and mounted in many public areas to support real-time monitoring. As they are operated continuously, they generate a massive amount of data contributing to big data. Among the other types of sources, CCTV data can be used as the foundation for accurate traffic flow estimation (Fedorov et al. 2019; Peppa et al. 2018).

The majority of recent research use state-of-the-art deep learning based object detection frameworks such as Faster R-CNN (Ren et al. 2015), YOLO (Redmon, Divvala, et al. 2016) and mask-RCNN (He, Gkioxari, et al. 2017) for vehicle detection and tracking (Laroca et al. 2018; Y. Zhang et al. 2017; F. Zhang et al. 2019; Q. Zhang et al. 2020) from image data sets. Faster R-CNN and mask-RCNN belong to the R-CNN family networks that use regions to locate the objects within an image. In comparison, the YOLO algorithm divides the entire image into cells and predicts the bounding boxes and class probabilities. However, traffic flow estimation using these algorithms for surveillance camera data sets is still in very early development. Difficulties in moving, storing, developing efficient, intelligent algorithms for processing and analyzing CCTV big data have been identified as major challenges. (Fedorov et al. 2019).

The study presented in this paper seeks to answer the questions; 1) What object detection algorithm is best suited to the CCTV image data set for vehicle detection? 2) Can traffic flow be estimated by counting the number of vehicles in CCTV images using an object detection algorithm?. Therefore, we collect real-time CCTV imagery from traffic cameras through the New Zealand Transport Agency's (NZTA) traffic cameras Application Programming Interface (API)[1]. During the first experiment, we compare the performance and accuracy of faster R-CNN, Mask R-CNN and YOLOv3 algorithms in vehicle detection for the CCTV image data set. Then, as a case study, we focus on one of the busiest roads in Christchurch Central Business District (CBD) to estimate the traffic flow. However, broader research extending this work would use the estimated flow of traffic to predict the short-term traffic flow. The results of this research, along with the prediction system can be used by city authorities to understand traffic flow patterns, predict traffic flow at a given time, understand traffic anomalies, and make management decisions.

The rest of our paper is outlined as follows. The next section reviews the existing work. Then we illustrate our methodology of the study and discuss the results. Finally, we present concluding remarks and future research steps.

RELATED WORK

One of the first steps in traffic flow estimation is vehicle identification. Object detection differs from classification as it attempts to draw a bounding box around the object of interest in order to locate it within the image. In computer vision research, there are three primary object detectors (Redmon, Divvala, et al. 2016; Ren et al. 2015):

- R-CNN and their variants, including the original R-CNN, Fast R- CNN, Faster R-CNN and Mask R-CNN
- Single Shot Detector (SSDs)
- YOLO

These Convolutional Neural Networks (CNN) based object detectors can be roughly divided into two main categories: single-stage detectors and two-stage detectors. The single-stage detectors are generally fast and predict object bounding boxes together with classes within a single network pass (e.g., SSDs and YOLO) (Redmon, Divvala, et al. 2016; Liu et al. 2016). Comparatively, two-staged detectors such as R-CNN family networks detection happens in two-stages; 1) the model proposes a set of regions of interests by selective search (Uijlings et al. 2013) or using Regional Proposal Network (RPN) 2) a classifier only processes the region candidates to identify the objects (Girshick et al. 2014; Girshick 2015; Ren et al. 2015; He, Gkioxari, et al. 2017). Therefore, two-stage detection tends to be slow. Huang et al. (Huang et al. 2017) provide a thorough review of the key advantages and disadvantages of single and two-stage detectors.

The use of CNNs to identify objects in all regions of an image has been computationally inefficient (Zhao et al. 2019). Therefore, Girshick et al. (Girshick et al. 2014) proposed the first Regions with CNN features (R-CNN) algorithm to use selective search (Uijlings et al. 2013) to extract just 2000 regions from images to propose candidate bounding boxes that could contain objects. The identified regions were then passed into a CNN for classification, which eventually led to one of the first deep learning-based object detectors.

However, the R-CNN algorithm was painfully slow and could not be used for real-time detection. Therefore, Girshick et al. (Girshick 2015) improved R-CNN and published a second paper in 2015, entitled Fast R- CNN. The

[1]NZTA Traffic Cameras API . Retrieved January 5, 2020, from https://www.nzta.govt.nz/traffic-and-travel-information/infoconnect-section-page/about-the-apis/traffic-cameras/

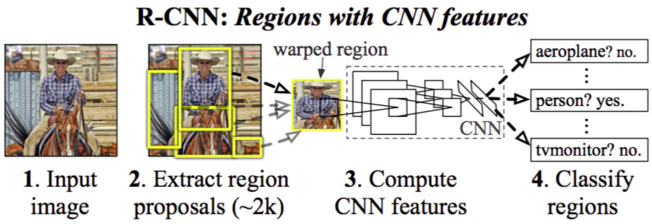

Figure 1. R-CNN architecture (Girshick et al. 2014).

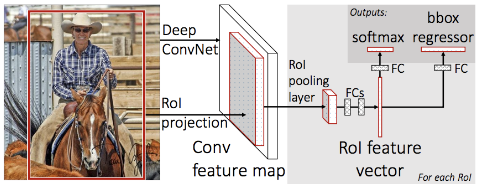

Figure 2. Fast R-CNN architecture (Girshick 2015).

Fast R-CNN algorithm made significant improvements to the original R-CNN, namely by increasing accuracy and reducing the time it took to perform the forward pass. However, the Fast-RCNN model still relied on an external region proposal algorithm (see Figures 1 and 2).

In 2015, the follow-up paper by Girshick et al. (Ren et al. 2015) introduced Faster R-CNN as a true end-to-end detector of deep learning artefacts. It has been improved by removing the selective search requirement and instead relying on RPN that is fully convolutional and can predict the object bounding boxes and "objectness" scores (e.g., a score that quantifies the probability of an image region may contain an object). The outputs from the RPNs were then passed into the R-CNN component for final classification and labelling.

Kaiming et al. (He, Gkioxari, et al. 2017) extended faster R-CNN by proposing instance segmentation rather than drawing bounding boxes, which resulted in proposing mask R-CNN. It is considered as a flexible and efficient framework compared to other R-CNN family networks (Zhao et al. 2019).

The most significant problem with the R-CNN family of networks was their speed as they were incredibly slow, even faster R-CNN obtaining only 5 FPS (Frame Per Second) on a Graphical Processing Unit (GPU) (Ren et al. 2015). Both SSDs and YOLO use a one-stage detector strategy to help increase the speed of deep learning object detectors. One-stage detectors treat object detection as a regression problem, taking a given input image and simultaneously learning bounding box coordinates and corresponding class label probabilities. Generally, single-stage detectors tend to be less accurate than two-stage detectors, but are significantly faster (Huang et al. 2017).

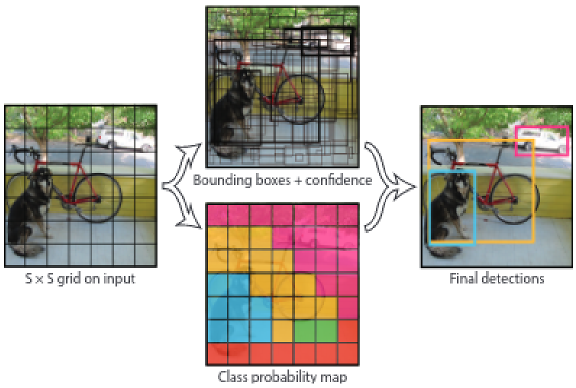

Figure 3. YOLO object detection (Redmon, Divvala, et al. 2016).

You Only Look Once (YOLO) was First introduced in 2015 by Redmon et al. (Redmon, Divvala, et al. 2016) as an object detector capable of real-time object detection, obtaining 45 FPS on a GPU (see Figure 3). YOLO has gone through several different iterations until they introduced YOLO9000 (Redmon and Farhadi 2017). They were able to achieve such a large number of object detection by performing joint training for both object detection and classification. The authors simultaneously trained YOLO9000 on both the ImageNet classification data set and the COCO detection data set using joint training. However, as the performance was not satisfactory, they recently introduced YOLOv3 (Redmon and Farhadi 2018), which is significantly larger than previous models and with greater accuracy. Table 1 provides the details of popular object detection models, their objectives and links to source codes.

Multiple open-source visual data sets with manually labelled features have contributed to the advancement in computer vision research (Gauen et al. 2017). We use YOLOv3 and R-CNN networks trained on two such data sets, namely Common Objects in Context (COCO) (Lin et al. 2014) and Computational Learning Visual Object

Table 1. Popular object detection models and their objectives

Algorithm	Author	Objective	Code
R-CNN	(Girshick et al. 2014)	Object detection	https://github.com/rbgirshick/rcnn
fast R-CNN	(Girshick 2015)	Object detection	https://github.com/rbgirshick/fast-rcnn
faster R-CNN	(Ren et al. 2015)	Real-time Object detection	https://github.com/rbgirshick/py-faster-rcnn
mask R-CNN	(He, Gkioxari, et al. 2017)	Image segmentation	https://github.com/facebookresearch/Detectron
YOLO, YOLO9000, YOLOv3	(Redmon, Divvala, et al. 2016; Redmon and Farhadi 2017; Redmon and Farhadi 2018)	Real-time object detection	https://pjreddie.com/darknet/yolo/

Classes (PASCAL VOC) (Everingham et al. 2015). COCO is an image data set introduced by Microsoft, consisting of 80 common objects in their natural context (Lin et al. 2014). Training and testing data sets for PASCAL VOC consisted of 27,450 detection objects in 11,530 images of 20 different classes. Also, the test and training data sets of PASCAL VOC segmentation consist of 6929 segmented objects in 11,530 images (Gauen et al. 2017; Everingham et al. 2015).

The *traffic flow estimation* is identifying the number of vehicles during the t^{th} time interval at the i^{th} observation location in a transportation network which can be denoted as X_i^t. Therefore, the *traffic flow prediction* problem can be stated as follows: Given X_i^t denote the observed traffic flow during the t^{th} time interval at the i^{th} observation location, $t = 1, 2, ..., T$ and $i = 1, 2, ..., m$, the problem is to predict the traffic flow at time interval $(t + \Delta)$ for some prediction horizon Δ (Lv et al. 2014). Most of the traffic-related research using Deep Learning techniques have focused mainly on the problem of traffic flow prediction (Yi et al. 2019; Polson and Sokolov 2017; Kunde et al. 2017). However, these studies are different from the scope of this paper. The main problem we address is the traffic flow estimation from CCTV image series as discussed by Fedorov at el. (Fedorov et al. 2019), a subject area in very early development. Therefore, there are a minimal number of researches which address the same problem.

Previously, inductive loop detectors, pneumatic road tubes, and temporary manual counts were the primary methods for estimating traffic flow (Barthélemy et al. 2019). However, depending on the costs and difficulties of installation, these methods can not be used in large areas. As highlighted in the introduction, CCTV monitoring is currently very common and these large networks are barely used except for the investigation of incidents and anti-social behaviour (Barthélemy et al. 2019). Previously, due to privacy concerns, only the police and city councils used to access these data. However, the current trend of most city councils is to put their CCTV data sets for open access. Fedorov et al. (Fedorov et al. 2019) have used Faster R-CNN for a video data set to identify traffic flow. However, only a short video clip containing 982 frames has been considered for their research. A similar study has evaluated the accuracy of two deep learning algorithms, namely MobileNet, and faster R-CNN trained on COCO data set (Peppa et al. 2018). They show that the accuracy of faster R-CNN is more for their CCTV image data set. However, the direction of vehicle movement is not considered during the flow estimation process. Taking advantage of the open access CCTV image series and considering the research gap, we focused on developing a method for estimating traffic flow.

METHODOLOGY

The study presented in this paper evaluates the performance of three Deep Learning algorithms and estimates the flow of traffic from CCTV images. Therefore, in the first experiment, we evaluate the performance and accuracy of three of the most popular object detection algorithms, such as faster R-CNN, mask R-CNN and YOLOv3. We used a faster R-CNN model trained on PASCAL VOC data set with ResNet-50 backbone (He, X. Zhang, et al. 2016) and mask R-CNN model trained on COCO data set with ResNet-50 backbone, which is implemented in GluonCV[2]. Also, YOLOv3 implementation in cvlib python library[3] trained on COCO data set. During the second experiment, we apply YOLOv3 to formulate the traffic flow by counting the number of vehicles in each minute. The flow of methodology is shown in Figure 4.

[2]GluonCV. Retrieved January 5, 2020, from https://gluon-cv.mxnet.io/index.html
[3]cvlib. Retrieved January 5, 2020, from https://www.cvlib.net/

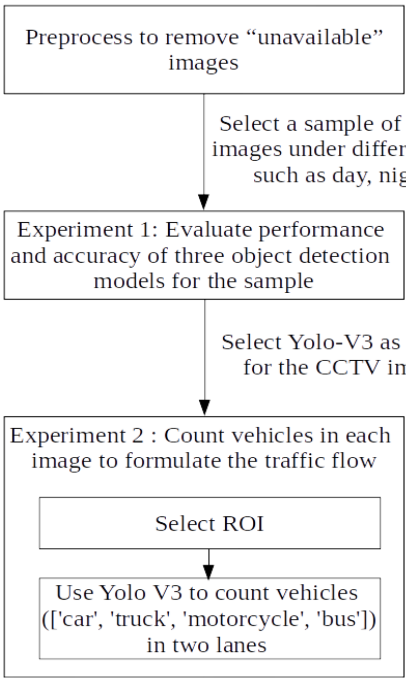

Figure 4. Methodology.

Data set

A CCTV image data set was formulated by collecting traffic camera images in real-time through the NZTA traffic cameras API from 10th of October to 31st of October 2019 in Christchurch CBD. There are 83 cameras operated in the CBD, and for this experiment, we selected a busiest road namely "West along Yaldhurst Rd from Curletts Rd" (latitude -43.53074, longitude 172.56812). The total size of the data set we selected for the experiments is 1.6 GB. The images are low resolution, taken from different angles, in different illumination levels and also under different weather conditions. Each image has a height of 600 pixels and a width of 800 pixels. A sample image is shown in Figure 5. Occasionally, cameras create an "unavailable" image with a message due to the technical faults (see Figure 6).

Figure 5. A sample CCTV image

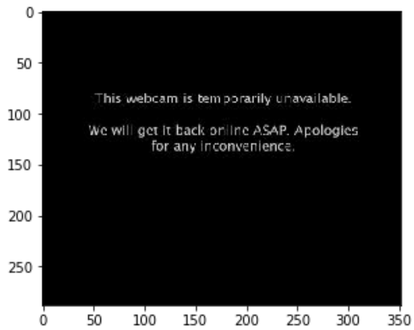

Figure 6. Camera "unavailable" image

Data processing

We found 6.3% of the data set considered for the experiment as "unavailable images" (see Table 2). Therefore, they were filtered out using a simple python script, considering the total pixel value.

Table 2. Data set before and after pre-processing

Total number of images before pre-processing	24, 085
Total number of “unavailable images”	1519
Total number of images after pre-processing	22, 566

Experiments

Experiment 1

We selected a sample 10% of the of pre-processed images under different conditions, such as day, night and blur. Then we evaluated the accuracy and performance of YOLOv3, mask R-CNN and faster R-CNN in the detection of vehicles in each of the classes such as ['car', 'truck', 'motorcycle', 'bus']. The performance was measured in terms of time taken to detect objects. According to Table 3, YOLOv3 has always achieved the highest performance by having the lowest time detect vehicles.

Table 3. Performance and accuracy of the three models for our CCTV data set

Model	Performance/ mean time taken to detect vehicles (seconds)	Recall	Precision
YOLOv3	0.86	0.79	0.96
faster R-CNN	8.37	0.50	0.96
mask R-CNN	55.6	0.69	0.77

For the same sample, we evaluated the accuracy in terms of precision and recall (see Table 3). First, we manually counted the number of vehicles in each image and then used the faster R-CNN, mask R-CNN and YOLOv3 models to count the number of vehicles (see Figure 7). Then precision and recall values were obtained. Precision is the number of True Positives (TP) over the number of predicted positives (PP), and recall is the number of true positives over the number of actual positives (AP). $PP = TP + FalsePositives\ (FP)$ and $AP = TP + FalseNegatives\ (FN)$ and therefore, Precision = $\frac{TP}{PP} = \frac{TP}{TP+FP}$ and Recall = $\frac{TP}{AP} = \frac{TP}{TP+FN}$.

Experiment 2

Our next experiment was to estimate the traffic flow by counting the number of vehicles in each image. Traffic flow estimation has two tasks; 1) determining the direction of movement of the vehicle 2) counting the number of vehicles in each image by the direction of the vehicle movement. Therefore, to identify the “left-lane” and the “right-lane”, each image must be divided into two. We also wanted to avoid the parking area, and hence, the most appropriate region of interest was identified as a trapezium. Two trapeziums were selected having the sizes points [[100, 600],[250, 199],[450, 199],[800, 600],[100, 600]] as the “left-lane” and [[750,600],[450,200],[800,200],[800,600],[100,600]] as the “right-lane” (see Figure 8 and 9). The algorithm for selecting the trapezium is as follows:

Algorithm 1 ROI selection as a trapezium

1: **for** $i \in I$ **do**
$y\_size, x\_size = i.shape[:2]$
$vert\_coef = 0.3333$
$hor\_coef = 0.312$
$v\_coef = vert\_coef$
$up\_left\_coef = hor\_coef$
$up\_right\_coef = 1 - up\_left\_coef$
$low\_left\_point = [0, y\_size]$
$low\_right\_point = [x\_size, y\_size]$
$up\_left\_point = [x\_size * up\_left\_coef, y\_size * v\_coef]$
$up\_right\_point = [x\_size * up\_right\_coef, y\_size * v\_coef]$
2: **end for**

We used the YOLOv3 implementation in Python cvlib to obtain the traffic flow count in every single minute.

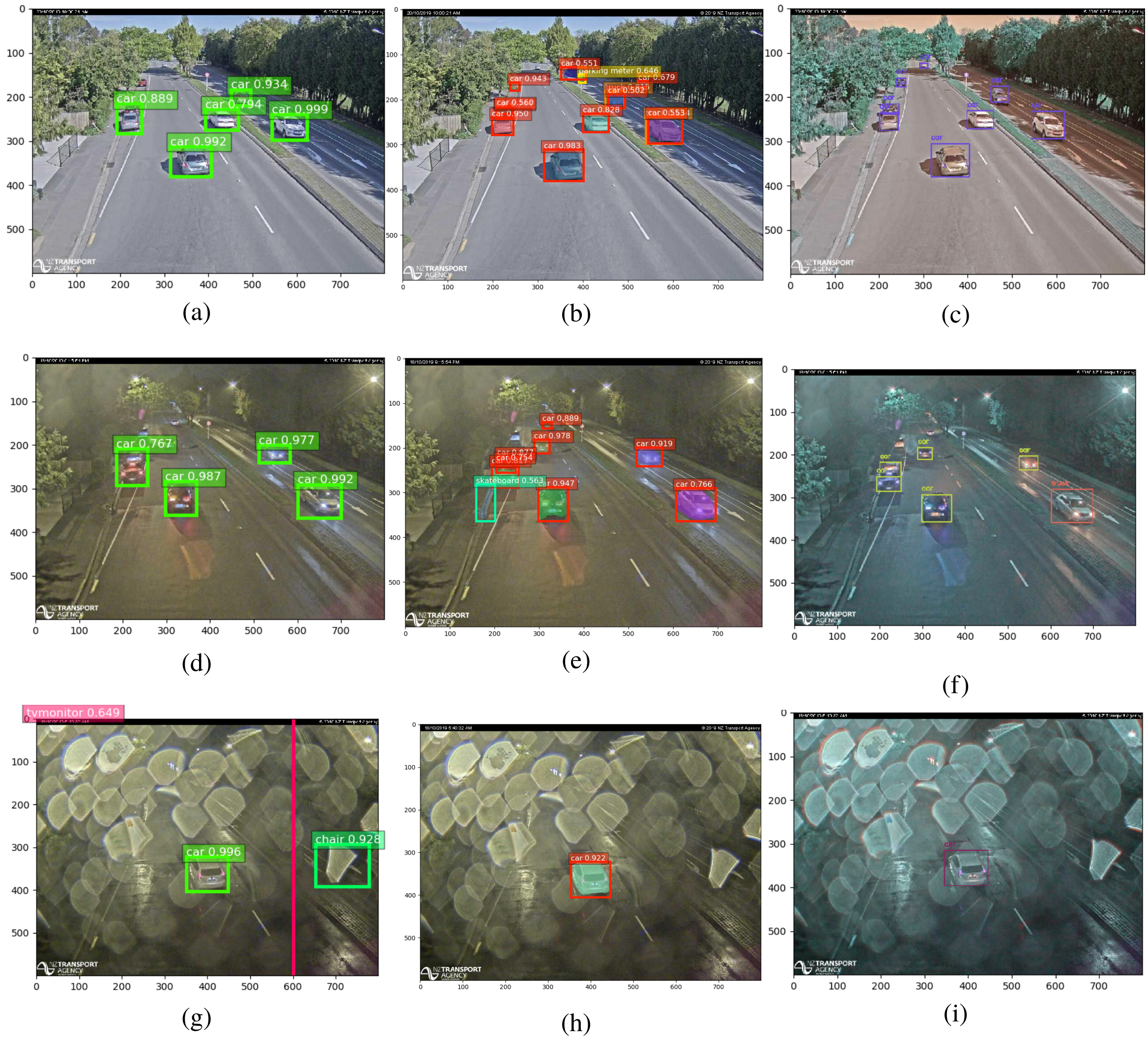

Figure 7. Vehicle Detection (a) faster R-CNN-Day (b) mask R-CNN-Day (c) YOLOv3 R-CNN-Day (d) faster R-CNN-Night (e) mask R-CNN-Night (f) YOLOv3-Night (g) faster R-CNN-Blur (h) mask R-CNN-Blur (i) YOLOv3-Blur

RESULTS AND DISCUSSION

According to Table 3, YOLOv3 has high recall and precision values for the selected data set. High precision relates to a low false-positive rate, and high recall relates to a low false-negative rate. High scores for both show that the model is returning accurate results. Comparatevely, faster R-CNN has high precision but low recall, which means that very few results are returned, but most of its identified objects are correct. furthermore, mask R-CNN returns a low recall and precision values compared to YOLOv3. Therefore, based on both performance and accuracy values, we have chosen YOLOv3 as the most appropriate algorithm for this project to estimate traffic flow. All experiments were carried on Ubuntu 18.04.3 with Nvidia Geforce graphics, 8 CPU cores (Intel(R) Core(TM) i7-8565U CPU @ 1.80GHz) and 8 GB RAM. Figure 10 and 11 show a sample of the vehicle counts generated by YOLOv3 for the data set. However, these results are not validated. Therefore, we will use a CCTV video recording at the same location to get the traffic flow by manually counting as the ground truth. Then, the generated traffic flow will be evaluated against the ground true flow.

The contribution of the paper can be summarized as follows:

- We have constructed a new, challenging data set by collecting CCTV images at each minute through the NZTA traffic cameras API, which includes a total of 24,085 images for the experiments discussed in this paper. To the best of our knowledge, this is the first time that such a large CCTV data set has been used to formulate traffic flow using Deep Learning.

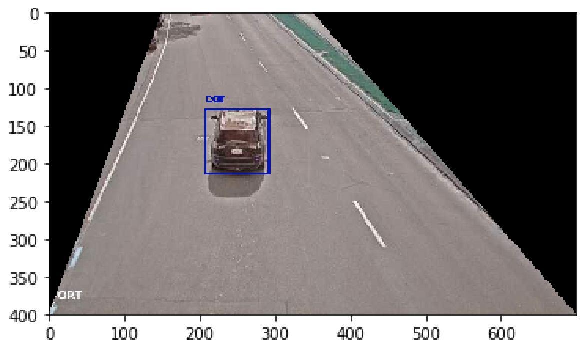

Figure 8. Left Lane.

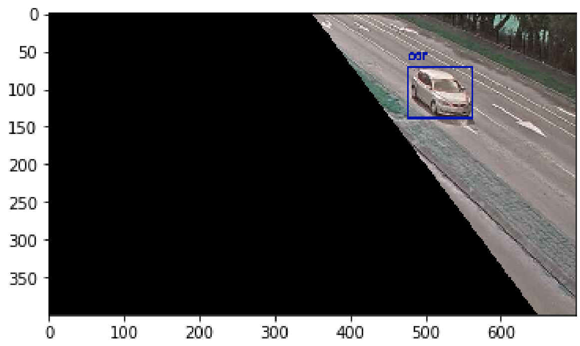

Figure 9. Right Lane.

Date	Time	LeftLane	RightLane
2019-10-15	14-44-00	1	0
2019-10-15	14-45-00	1	0
2019-10-15	14-46-00	0	4
2019-10-15	14-47-00	0	4
2019-10-15	14-48-00	0	4
2019-10-15	14-49-00	0	0
2019-10-15	14-50-00	0	0
2019-10-15	14-51-00	1	10
2019-10-15	14-52-00	1	10
2019-10-15	14-53-00	1	5

Figure 10. A sample of the obtained vehicle counts

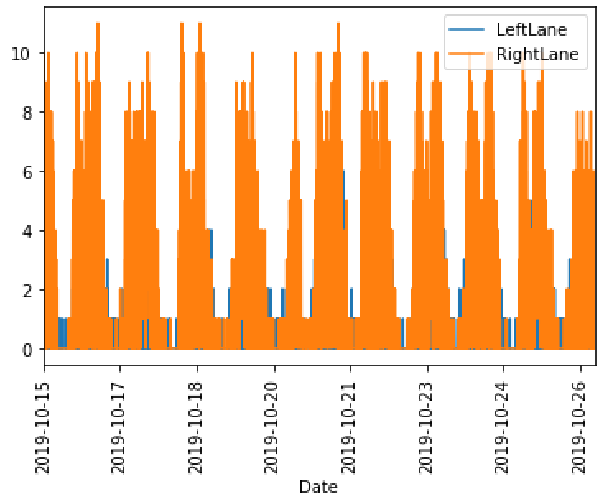

Figure 11. A plot showing a sample of traffic flow

- We have evaluated the performance and accuracy of YOLOv3, faster R-CNN and mask R-CNN in counting vehicles for the CCTV images. Then, we obtained the traffic flow counts for the selected road at Christchurch CBD.
- We have developed a simple ROI algorithm to identify “left-lane” and “right-lane” in the CCTV images to identify the direction of vehicle movement.

CONCLUSION

In this paper, we discussed the problem of traffic flow estimation as the first step of a broader project for emergency traffic management. As a case study, we considered one of the busiest roads in Christchurch, New Zealand. This subject area is still in the early stages of development, and there are only a few works that aim to process the CCTV image series automatically for traffic analysis. To address this issue, we started by evaluating the performance accuracy of three popular object detection models, namely faster R-CNN, mask R-CNN and YOLOv3. Our experiment 1 results showed that YOLOv3 was very fast to detect objects compared to the other two models. Also, in the same experiment, we demonstrated that YOLOv3 had the highest accuracy. During the experiment 2, we introduced a simple ROI selection heuristic algorithm to select “left-lane” and “right-lane” of each image. We applied the YOLOv3 model to count the number of vehicles in such as car, bus, truck and motorcycle in each minute to formulate the traffic flow.

Future work for this project beyond this paper will be extended to the identification of traffic flow to the entire city of Christchurch using multi-cameras. Then, the short-term traffic flow will be predicted at any location in the city. During the final step of the project, the prediction of traffic flow will be compared with the real-time traffic flow to identify traffic anomalies. This will allow emergency management personnel to decide whether to re-route, change traffic signals or make any other decisions within a few seconds of an incident. Also, during an incident, emergency managers can use the prediction to estimate the traffic flow attempting to evacuate from different routes in the city.

REFERENCES

Barthélemy, J., Verstaevel, N., Forehead, H., and Perez, P. (2019). "Edge-computing video analytics for real-time traffic monitoring in a smart city". In: *Sensors* 19.9, p. 2048.

Carpender, S. K., Campbell, P. H., Quiram, B. J., Frances, J., and Artzberger, J. J. (2006). "Urban evacuations and rural America: lessons learned from Hurricane Rita". In: *Public Health Reports* 121.6, pp. 775–779.

Everingham, M., Eslami, S. A., Van Gool, L., Williams, C. K., Winn, J., and Zisserman, A. (2015). "The pascal visual object classes challenge: A retrospective". In: *International journal of computer vision* 111.1, pp. 98–136.

Fedorov, A., Nikolskaia, K., Ivanov, S., Shepelev, V., and Minbaleev, A. (2019). "Traffic flow estimation with data from a video surveillance camera". In: *Journal of Big Data* 6.1, p. 73.

Gauen, K., Dailey, R., Laiman, J., Zi, Y., Asokan, N., Lu, Y.-H., Thiruvathukal, G. K., Shyu, M.-L., and Chen, S.-C. (2017). "Comparison of visual datasets for machine learning". In: *2017 IEEE International Conference on Information Reuse and Integration (IRI)*. IEEE, pp. 346–355.

Girshick, R. (2015). "Fast r-cnn". In: *Proceedings of the IEEE international conference on computer vision*, pp. 1440–1448.

Girshick, R., Donahue, J., Darrell, T., and Malik, J. (2014). "Rich feature hierarchies for accurate object detection and semantic segmentation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580–587.

Harten, E., Patel, R., Ryan, L., Sawyer, J., Barbera, J., Mazzuchi, T., and Santos, J. (2018). "Evaluation of traffic mitigation strategies for pre-hurricane emergency evacuations". In: *2018 Systems and Information Engineering Design Symposium (SIEDS)*. IEEE, pp. 214–219.

He, K., Gkioxari, G., Dollár, P., and Girshick, R. (2017). "Mask r-cnn". In: *Proceedings of the IEEE international conference on computer vision*, pp. 2961–2969.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778.

Huang, J., Rathod, V., Sun, C., Zhu, M., Korattikara, A., Fathi, A., Fischer, I., Wojna, Z., Song, Y., Guadarrama, S., et al. (2017). "Speed/accuracy trade-offs for modern convolutional object detectors". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7310–7311.

Kang, D., Lv, Y., and Chen, Y.-y. (2017). "Short-term traffic flow prediction with LSTM recurrent neural network". In: *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, pp. 1–6.

Kunde, F., Hartenstein, A., Pieper, S., and Sauer, P. (2017). "Traffic Prediction using a Deep Learning Paradigm." In: *EDBT/ICDT Workshops*.

Laroca, R., Severo, E., Zanlorensi, L. A., Oliveira, L. S., Gonçalves, G. R., Schwartz, W. R., and Menotti, D. (2018). "A robust real-time automatic license plate recognition based on the YOLO detector". In: *2018 International Joint Conference on Neural Networks (IJCNN)*. IEEE, pp. 1–10.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). "Microsoft coco: Common objects in context". In: *European conference on computer vision*. Springer, pp. 740–755.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. C. (2016). "Ssd: Single shot multibox detector". In: *European conference on computer vision*. Springer, pp. 21–37.

Lv, Y., Duan, Y., Kang, W., Li, Z., and Wang, F.-Y. (2014). "Traffic flow prediction with big data: a deep learning approach". In: *IEEE Transactions on Intelligent Transportation Systems* 16.2, pp. 865–873.

Peppa, M., Bell, D., Komar, T., and Xiao, W. (2018). "URBAN TRAFFIC FLOW ANALYSIS BASED ON DEEP LEARNING CAR DETECTION FROM CCTV IMAGE SERIES." In: *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences* 42.4.

Polson, N. G. and Sokolov, V. O. (2017). "Deep learning for short-term traffic flow prediction". In: *Transportation Research Part C: Emerging Technologies* 79, pp. 1–17.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). "You only look once: Unified, real-time object detection". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788.

Redmon, J. and Farhadi, A. (2017). "YOLO9000: better, faster, stronger". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7263–7271.

Redmon, J. and Farhadi, A. (2018). "Yolov3: An incremental improvement". In: *arXiv preprint arXiv:1804.02767*.

Ren, S., He, K., Girshick, R., and Sun, J. (2015). "Faster r-cnn: Towards real-time object detection with region proposal networks". In: *Advances in neural information processing systems*, pp. 91–99.

Uijlings, J. R., Van De Sande, K. E., Gevers, T., and Smeulders, A. W. (2013). "Selective search for object recognition". In: *International journal of computer vision* 104.2, pp. 154–171.

Yi, H., Bui, K.-H. N., and Jung, H. (2019). "Implementing A Deep Learning Framework for Short Term Traffic Flow Prediction." In: *WIMS*, pp. 7–1.

Yuan, S., Ma, Y., Zhang, H., and Liu, Y. (2013). "An urban traffic evacuation model with decision-making capability." In: *ISCRAM*.

Zhang, F., Yang, F., and Li, C. (2019). "Fast vehicle detection method based on improved YOLOv3 [J]". In: *Computer Engineering and Applications* 55.2, pp. 12–20.

Zhang, Q., Chang, X., and Bian, S. B. (2020). "Vehicle-Damage-Detection Segmentation Algorithm Based on Improved Mask RCNN". In: *IEEE Access* 8, pp. 6997–7004.

Zhang, Y., Wang, J., and Yang, X. (2017). "Real-time vehicle detection and tracking in video based on faster R-CNN". In: *Journal of Physics: Conference Series*. Vol. 887. 1. IOP Publishing, p. 012068.

Zhao, Z.-Q., Zheng, P., Xu, S.-t., and Wu, X. (2019). "Object detection with deep learning: A review". In: *IEEE transactions on neural networks and learning systems*.