Electric Scooter and Its Rider Detection Framework Based on Deep Learning for Supporting Scooter-Related Injury Emergency Services

Hoa Nguyen(✉), Minh Nguyen(✉), and Qian Sun(✉)

Auckland University of Technology, Auckland, New Zealand
{h.nguyen,minh.nguyen}@aut.ac.nz

Abstract. Electric scooters (e-scooters) have been considered as a "last mile" solution to existing public transportation systems in many cities all over the world due to their convenience at a highly affordable price. E-scooters enable users to travel distances which are too long to walk but too short to drive, so they help to reduce the number of cars on the roads. Along with its increasing popularity, accidents involving e-scooters have become a growing public concern, especially in large cities with heavy traffic.

It is useful to detect and include e-scooters in traffic control. However, there is no available pre-built-model for detecting an electric scooter. Therefore, in this paper, we proposed a scooter and its rider detection framework that supports emergency management for scooter-related injuries. The framework helps to identify scooter and its rider in live-stream videos and can be applied in traffic incidents detection applications. Our model was developed based on deep learning object detection models. Using ImageAI API, we trained and deployed our own model based on 200 images acquired on the internet. The preliminary results appeared to be robust and fast; however, the accuracy of our proposed model could be improved if using a larger dataset for training and evaluating.

Keywords: Object detection · Deep learning · Electric scooter · Emergency services

1 Introduction

1.1 Motivation

Electric scooters (e-scooters), promoted by sharing companies, first launched in San Francisco in 2017, and then expanded as a "last mile" solution to existing public transportation systems in many cities all around the world [1,2]. Reports from the two most popular companies, Lime and Bird, showed that each has more than 10 million rides since their first operation in July and September

M. Nguyen et al. (Eds.): ISGV 2021, CCIS 1386, pp. 233–246, 2021.
https://doi.org/10.1007/978-3-030-72073-5_18

2017, respectively [3]. Not only offering an attractive travelling method for too long distances to walk but too short to drive, e-scooters may help to reduce the number of cars on the roads as well. However, the increasing popularity of e-scooters has raised public safety concerns to riders and pedestrians [4].

There have been numerous reports of e-scooter-related injuries involving its riders, pedestrians and other road vehicles. In Singapore, around 90 crashes related electric devices including e-scooters have been reported in the first half of 2017, resulting in about 90 injuries and 4 deaths [5]. A study conducted by Trivedi et al. [6] reported that 249 patients presented to 2 urban emergency departments in Southern California with injuries associated with standing electric scooter use during the study period from September 2017 to August 2018. In Australian countries, an increased burden on healthcare services for e-scooter related injuries has been seen [1]. A study performed at a single emergency department in Brisbane showed that 78% of e-scooter patients required X-rays and 24% required CT scans [7]. While over 200 plain films and 47 CT scans were requested for e-scooter related injuries at Auckland City Hospital in the first 2 months after e-scooters have been launched [8]. Unsafe riding behaviours, such as going into unauthorized traffic lanes, not wearing helmet, or overloading the scooter with multiple riders, etc., may be associated with the presentations for e-scooter related injuries [9].

Thanks for the development and progress of science and technology, object detection based on deep learning has been widely applied in various applications, such as driver-less car, robotics, video surveillance and pedestrian detection [10]. The technology is capable of assisting people to efficiently solve traffic problems by detecting vehicles in the traffic.

1.2 Proposed Idea

In this paper, we propose an e-scooter and its rider detection framework based on deep learning as illustrated in Fig. 1. There are four loosely connected components: camera system, YouTube server, processing computer, and control centre. The camera system captures activities from real-life and then broadcasts the stream to YouTube server. Processing computer downloads data from YouTube server and sends the broadcast results back to server. Afterwards, the detection result of the traffic flow including e-scooters, riders, vehicles as well as pedestrians will be displayed on the computer. Finally, the computer sends the results and notification to the control centre which provides the analysis and statistics for supporting decision making, or generates alarm in case the system detects a rider falling from scooter. The system helps detect incidents when e-scooters are used in the traffic. For example, from the auto-detection, we can detect the number of people riding on the e-scooter, whether the scooter goes into unauthorized traffic lanes or not, or even whether there is any falling from the e-scooter or not. Risk situations are reported and alarms will be sent to emergency services if an incident is detected. The custom model that can achieve scooter and its rider detection was trained by 200 images of scooters and its rider dataset which was collected from the internet. It has the ability to detect both scooter and its rider with background noise.

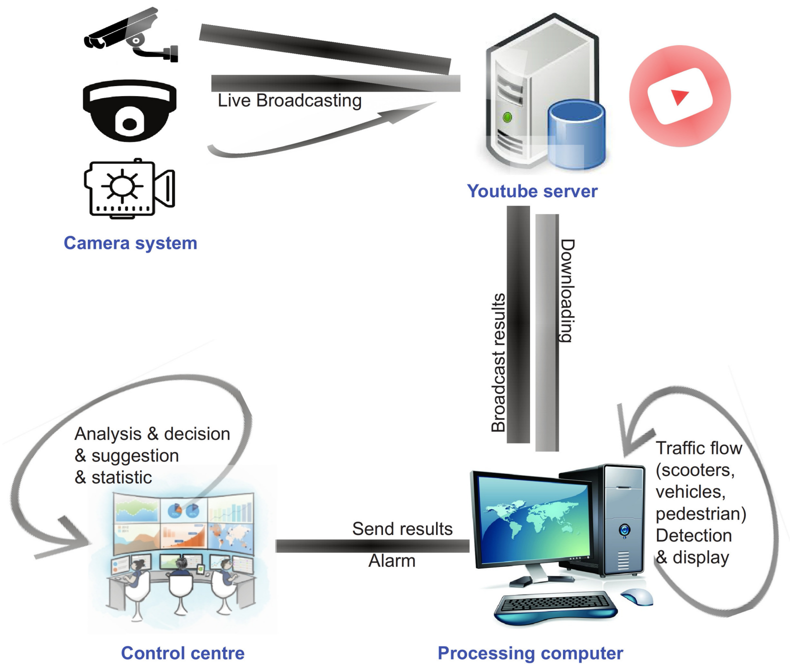


Fig. 1. Overall diagram of our proposed intelligent video-surveillance framework

2 Background

With the development and progress of science and technology, repetitive and time-consuming work have been taken over by the computer. Computer vision, as an inter-discipline based on image processing, machine learning and pattern recognition, is a rapidly developing research field in recent years. Object detection is a significant task in computer vision and it is used in detecting an object from certain scenes via some specific approach and algorithm [11]. One of the common application of object detection for public problem is to detect vehicles in the traffic. This section summarized traditional object detection methods as well as deep learning-based algorithms of object detection which is the background for building our deep learning model.

2.1 Traditional Object Detection Algorithms

In earlier studies, various artificially extracted features were employed for object detection. The three most commonly used features were Haar [12], HOG [13], and LBP [14]. The classifiers matched by these three features which mainly include the support vector machine (SVM), Adaboost, and the Haar feature set combined with the Adaboost. For many years, they have been widely used in the computer vision methods, initially intended for face detection [15]. The

HOG feature combined with the SVM classifier, has been widely used in image recognition, and it has achieved great success in pedestrian detection.

2.2 Deep Learning-Based Object Detection Algorithms

In recent years, deep learning methods has been becoming a hot research topic for many researchers, and a large number of deep learning target detection algorithms have been developed. Compared with traditional methods, deep learning method requires a massive amount of data, and automatic learning can reflect the characteristics of data differences, hence, makes it more representative. In the meantime, in visual recognition, the process of a Convolutional Neural Network (CNN) layer feature extraction is similar to the human visual mechanism, which represents the process from the edge to a part to the whole [16]. Recently, the deep learning target detection algorithms have obtained competitive real-time performance compared with the traditional methods due to the continuous expansion in data volume and rapid update of devices hardware. Thus, it has begun to gain recognition from the industry worldwide.

R-CNN: The first R-CNN algorithm was proposed by Ross Girshick in 2014, he combined region proposals with CNNs to achieve scale object detection [18]. The R-CNN algorithm performs the region search first, and then classifies the candidate regions. The Selective Search Method of R-CNN is used to generate candidate regions [18]. Its outstanding feature is to reduce information redundancy and increase detection speed.

Fast R-CNN: Fast R-CNN, an upgraded network structure based on both of R-CNN and SPP-Net, was proposed by Girshick in 2015 [19]. It applied Softmax classifier instead of SVM classifier so that achieving end-to-end training with a multi-task loss rather than single-stage. Training is able to update all network layers, in order to get higher detection accuracy, which is the most advantage of the algorithm. The RoI pooling is the core algorithm module. Max pooling is being used in the RoI pooling layer to transfer the features inside all valid area where is related into a smaller feature map along with a static spatial range of $H \times W$. The H and W are the layer hyper-parameters which are isolated from any specific RoI [19].

Faster R-CNN: Faster R-CNN, such a kind of optimization Fast R-CNN, has saved more the running time of detection networks than Fast R-CNN [20]. Moreover, it is a single, unified network for object detection. Because it introduces a concept of the Region Proposal Network (RPN), which will generate region proposals from neural network used to learn by itself. RPN is a fully-convolutional network that is naturally implemented, which support being trained end-to-end by back-propagation as well as stochastic gradient descent. Any size of the image will be taken as input by the RPN. Then the RPN outputs a set of rectangular object proposals which has an objectness score [20].

YOLO: YOLO, an abbreviation for You Look Only Once, is a new algorithm for object detection. Compared to previous work in the related fields, YOLO mount object detection as a spatial regression problem to isolated bounding boxes and related class probabilities. It is a single neural network which can predict bounding boxes and class probabilities directly from full images in one single evaluation process [21]. This algorithm is capable of predicting what type of objects are included and where they are in the frame. It has three significant advantages than other object detection algorithms. Firstly, it performs very fast due to its instinct for the regression problem without a complex pipeline. Secondly, it executes the reasoning via the global approach when performing prediction in which leads to less background errors. Finally, YOLO learns the data from a more generic representation which helps to learn easily in another domain [21]. Whereas, YOLO still have some limitations of object detection. YOLO limits the number of adjacent objects that our model can predict due to the spatial restriction. The spatial restriction is due to each grid cell can only predict two boxes with only one class [21]. Therefore, the issue will occur with small objects coming up as groups.

YOLOv3: YOLOv3 is a better solution compared to YOLO as it takes the anchor box idea from Faster R-CNN. YOLOv3 makes it better to detect small objects due to using the Darknet53 network which is based on the residual network as a feature extractor, and thus multi-scale detection and multi-label classification are the improvements of YOLOv3 [22]. Furthermore, YOLOv3 gets rid of the manually selected anchor box and execute k means clustering based on the dimension of the bounding boxes to achieve a better prior [23]. Especially in the aspect of handling the issue of the same target having two labels, Sigmoid function as the activation function for class prediction has better performance than softmax function [23]. As a result, YOLOv3 enable to perform multi-tag classification of a specific target via binary cross-entropy loss and logistic regression during the training process [22].

3 Design and Implementation

Figure 1 illustrates the overall diagram of our proposed framework. In this paper, we mainly focus on training a deep learning model for detecting electric scooter and its rider.

3.1 E-Scooter and Its Rider Detection Framework

Our proposed framework for detecting e-scooter and its rider is illustrated in Fig. 2. By default, the camera system captures images from real-life and sends the stream to Youtube server. The processing computer downloads the data from YouTube server and runs custom object detection algorithm. Here YOLOv3 object detection model is applied to detect the object. However, using the official object detection model (YOLOv3), our system can only recognize a number of

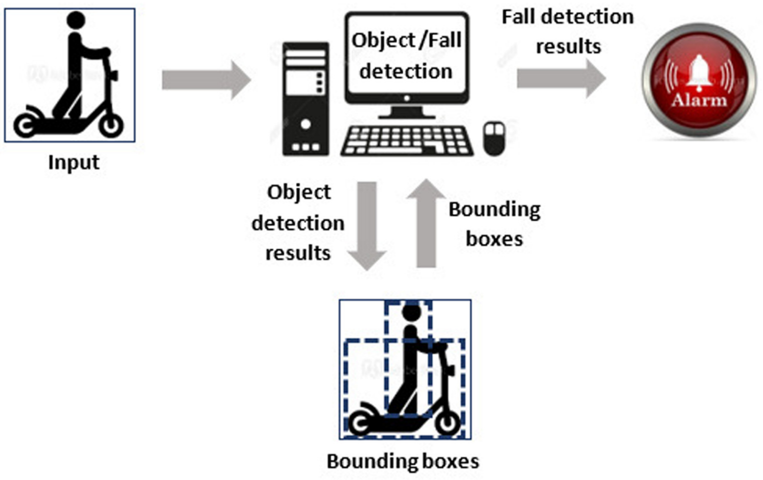


Fig. 2. Proposed e-scooter and its rider detection framework

pre-defined classes of objects. A new model was trained by transfer learning to achieve recognition for e-scooter and its rider. Object detection application provides bounding boxes of the objects including scooters, riders (person with scooter) and pedestrians. Bounding boxes of riders are considered as input of fall detection application. If it detects a fall, it generates an alarm and sends to control centre. Algorithm 1 summarizes object detection process and Algorithm 2 defines fall detection process. The pseudocode for Algorithm 1 and Algorithm 2 are below.

Algorithm 1. Pseudo code for custom object detection

```
Input: Data (video stream) from YouTube.
Output: Bounding boxes.
Method:
 1: Receive data (video stream) from YouTube.
 2: Run custom object detection model.
 3: Save bounding boxes into xml file.
 4: When End of data ⇒ Exit.
```

Figure 3 shows how the model works based on YOLOv3 algorithm. Our custom model forms the model of detection as a regression perspective. It separates the image into an even grid and parallel calculates bounding boxes with its confidence and class probabilities.

Algorithm 2. Pseudocode code for fall detection

```
Input: Bounding boxes (scooters, riders) from xml file.
Output: Fall detection results (1: Fall, 0: Not fall).
Method:
    Receive data (bounding boxes) from xml file.
 2: Run custom fall detection model.
    Save fall detection results into xml file.
 4: If Fall then send alarm to control centre.
    When End of data ⇒ Exit.
```

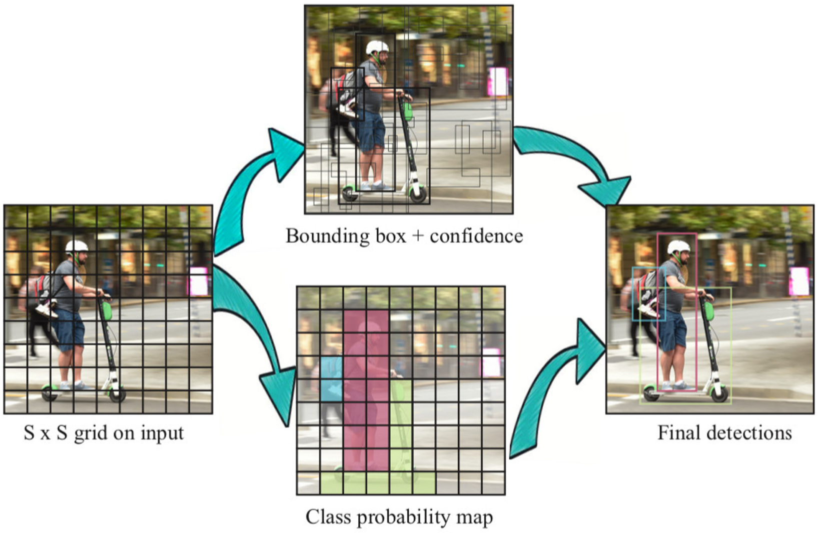


Fig. 3. E-scooter and its rider detection model based on YOLO

3.2 E-Scooter and Its Rider Detection Model Training and Deploying

The task of e-scooter and its rider detection model training and deploying includes the following parts.

Prepare Dataset: 200 images are collected from Google Image and Baidu Image using keyword "rider and scooter". We take the first 140 (70%) images as the training set and take the rest 60 (30%) images as the validation set. We use the labelling tool to select the person area and the scooter area at the image respectively and then save it to the annotation xml file. We repeat this process 200 times. After labelling the images, the folder structure of the dataset folder includes training dataset and validation dataset.

Install ImageAI and Dependencies: The processing computer used in this work is a non-GPU machine - a MacBook Air machine with Intel Core i5 processor @ 1.6 GHz and 4.00 GB of RAM, running macOS Mojave operating. Since our local Mac Book does not have GPU, Google colab is the best way to run the training for the new model. ImageAI is used for high-level training the new deep learning network and it can be installed via *"!pip3 install imageai –upgrade"*. Tensorflow is installed via *"!pip3 install tensorflow-gpu==1.13.1"* as the lower level dependency of ImageAI.

Train Custom Model: For achieving the goal, Google Colab, a free cloud service, is utilized to train the model. The custom model training uses the ImageAI *DetectionModelTrainer* API. This API is based on YOLOv3 algorithm, which is for unified, real-time object detection. Processing images via YOLOv3 is simple and straightforward to shorten detection time. In order to achieve a better detection accuracy, transfer learning from a pre-trained YOLOv3 model will be applied in the training.

Evaluate Model: In evaluation step, we import the DetectionModelTrainer class and create an instance of it. Then we set the model type to YOLOv3 and set the data directory as well as pass the path of files including h5 file, json file as the *"model_path"* and the *"json_path"*.

Test Model: We use the *CustomVideoObjectDetection* API from ImageAI to test our custom scooter and its rider detection model. Firstly, we import the *CustomVideoObjectDetection* class from ImageAI and the os. Then we instantiate a new instance from the *CustomVideoObjectDetection* class and set the model type to YOLOv3 (pre-trained transfer model). After that, we load the trained model and pass the path of the input video and the path of the output video created by ImageAI.

3.3 Fall Detection Implementation

Fall detection process is illustrated in Fig. 4. From the person bounding box detected by object detection model, we identify the centre point of the box and track this point. We draw the movement of the centre point in continuous detected bounding boxes. Standing and lying states are defined based on the aspect ratio of a bounding box (HW ratio), which is the ratio of the box's height to the box's width [28]. If height is larger than width, it is standing state. If height is smaller than width, it is lying state. The distance between centres of points of bounding boxes after each fall is used to verify real falls away from all other actions (include false positive). Fall event notification is only activated when the detected human does not show to be able to stand up after the fall.

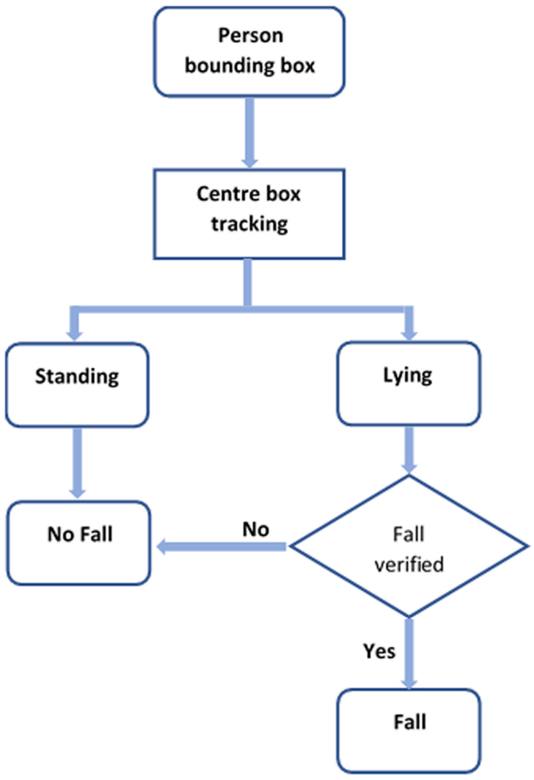


Fig. 4. Fall detection process

4 Results and Evaluations

4.1 Training Model Process

Once the object is detected, the generated output video will contain boxes with percentage probabilities numbers on the object detected from ImageAI. A screenshot of e-scooter and its rider detection is shown by Fig. 5

During the custom scooter and its rider detection training process, the value of both *"batch_size"* and *"epoch"* are the significant factors to determined the detection accuracy, due to the loss value will be influenced.

The result of the Fig. 6 shows that the overall trend is that as numbers of epoch increases, the loss decreases. The significant drop of the loss is after the first two epoch. Comparing the batch size 4 and batch size 8, the batch size 4 has less loss than size 8 overall.

The result of this accuracy vs epoch chart shows demonstrates the difference between running the training against batch size 8 and 4 in Fig. 7. As shown in the Y axis, the accuracy increases slowly after the 3rd epoch. Consequently, the figure shows that the size of batch 4 achieves better accuracy than the size 8.

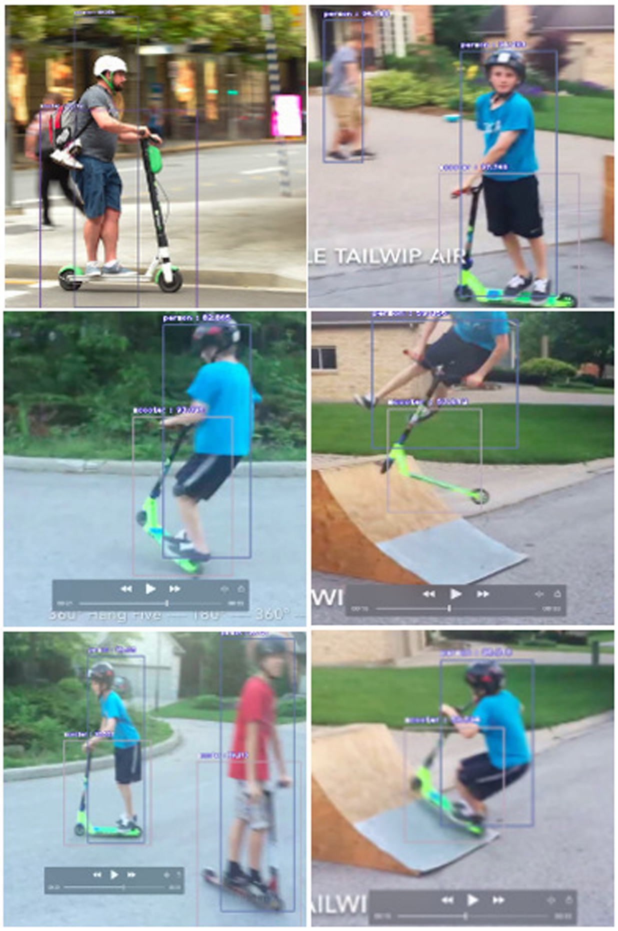


Fig. 5. E-scooter and its rider detection results

4.2 Evaluation Model Process

We mainly use three threshold values to control the process of the evaluation during the implementation stage of scooter and its rider detection. Those three thresholds are *"iou_threshold"*, *"object_threshold"* and *"nms_threshold"*.

– **"iou_threshold":**
  "iou_threshold", is the value of minimum Intersection over Union (IoU). It can set from 0.0 to 1.0. The IoU score is a common benchmark score for semantic segmentation [25]. It is being used widely in the area of computer vision due to the PASCAL VOC segmentation challenges which is a benchmark for visual object category detection and recognition. It supplies a standard dataset of images with their annotation as well as the standard evaluation processes for machine learning [26].

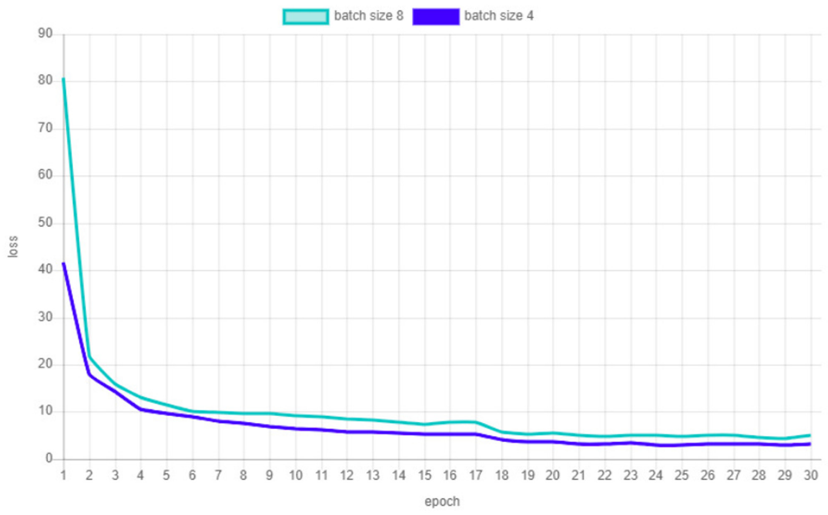


Fig. 6. The loss of batch size 4 and 8

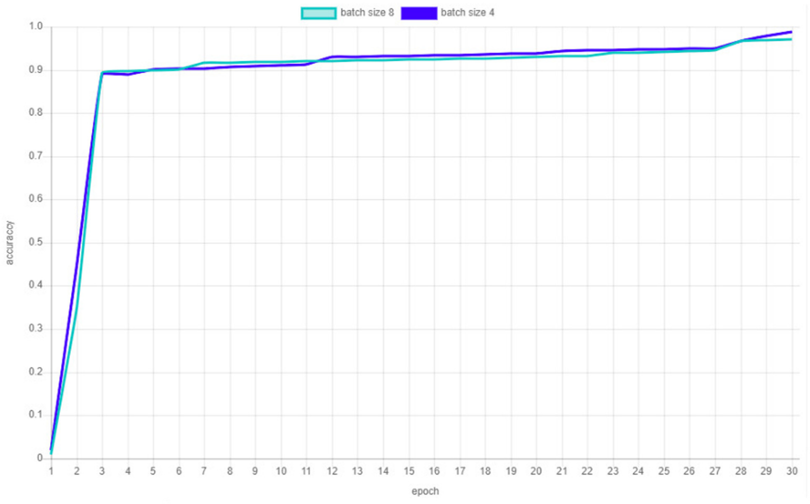
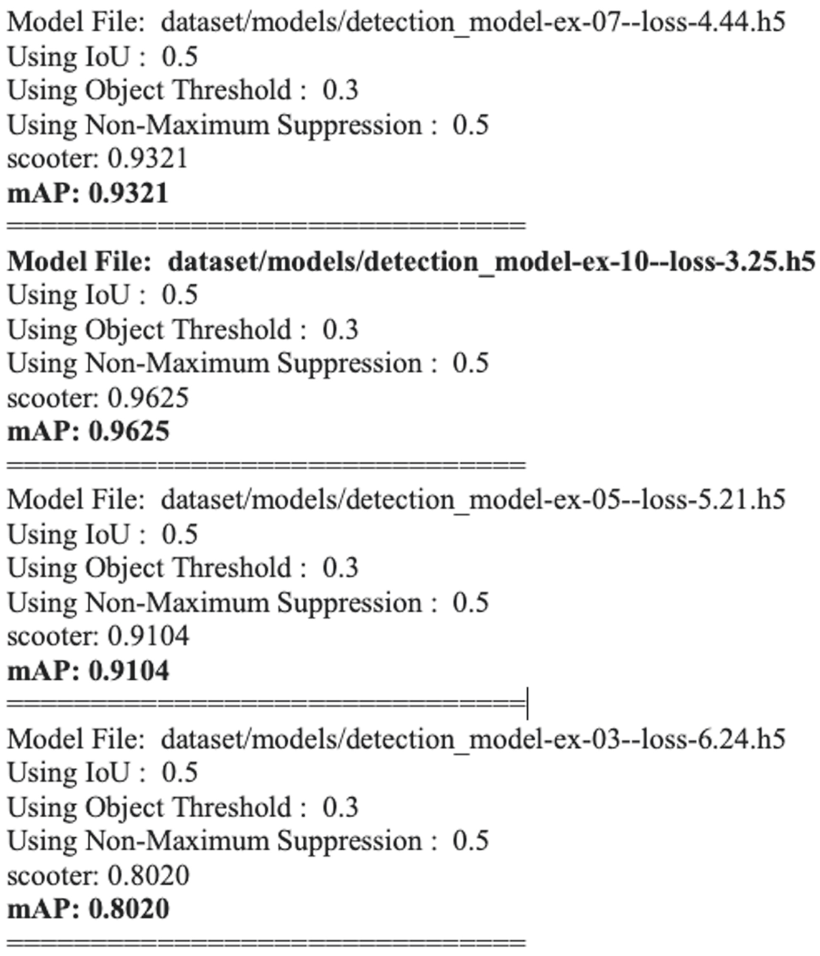


Fig. 7. The accuracy of batch size 4 and 8

- **"Object_threshold":**
  "Object_threshold" is another parameter of minimum class score for the mAP computation.
- **"nms_threshold":**
  "nms_threshold", is the Non-maximum suppression(NMS) for the mAP computation. NMS is a post-processing algorithm accountable for combining all detections which all belong to the same object [27].

In the context of this experiment, both the *iou* and *nms* set to 0.5, while the *object threshold* is set to 0.3. After succesffully running the evaluation code, the evaluation results of the model is shown in Fig. 8. From the evaluation results, taking mAP and loss value into accounts, it obviously shows that in most cases, the accuracy of the model increases when mAP increases or the loss decreases.



```
Model File:  dataset/models/detection_model-ex-07--loss-4.44.h5
Using IoU :  0.5
Using Object Threshold :  0.3
Using Non-Maximum Suppression :  0.5
scooter: 0.9321
mAP: 0.9321
===============================
Model File:  dataset/models/detection_model-ex-10--loss-3.25.h5
Using IoU :  0.5
Using Object Threshold :  0.3
Using Non-Maximum Suppression :  0.5
scooter: 0.9625
mAP: 0.9625
===============================
Model File:  dataset/models/detection_model-ex-05--loss-5.21.h5
Using IoU :  0.5
Using Object Threshold :  0.3
Using Non-Maximum Suppression :  0.5
scooter: 0.9104
mAP: 0.9104
===============================|
Model File:  dataset/models/detection_model-ex-03--loss-6.24.h5
Using IoU :  0.5
Using Object Threshold :  0.3
Using Non-Maximum Suppression :  0.5
scooter: 0.8020
mAP: 0.8020
===============================
```

Fig. 8. Evaluation results

5 Conclusion and Future Work

With the rapid development and progress of science and technology, especially in the area of computer vision research, object detection is the main hot topic via deep learning. This technology can apply to different scenarios to make our life becoming more intelligent. One of the common applications of object detection is to detect vehicles in the traffic via deep learning model. It is widely applied in assisting people to efficiently solve traffic problems.

As e-scooters have become more and more popularity in many large cities all around the world, detecting and including e-scooters in traffic control are useful. These applications may help to support safe e-scooter riding behaviours as well as provide timely emergency intervention if incidents related to e-scooters happen, and then may eliminate bad affect of e-scooter related injuries.

This paper focused on applying deep learning models to detect e-scooter and its rider in the traffic. The aim of this research is to support risk situations awareness and timely emergency management if incidents happen. Due to the limitation of the hardware, we only used 200 images for training and evaluating the scooter and its rider detection model. Results from running with maximum 30 epochs shows that the accuracy is acceptable to detect objects in images and videos.

The future work involves developing fall detection model, training and testing our proposed framework with a larger dataset to improve the accuracy of our model. Furthermore, we will test our proposed system on live streaming videos to evaluate its capability of applying in real applications.

References

1. Beck, S., Barker, L., Chan, A., Stanbridge, S.: Emergency department impact following the introduction of an electric scooter sharing service. Emerg. Med. Australas. **32**(3), 409–415 (2020)
2. Sipe, N., Pojani, D.: Can e-scooters solve the ‘last mile’ problem? They’ll need to avoid the fate of dockless bikes (2018)
3. Aizpuru, M., Farley, K.X., Rojas, J.C., Crawford, R.S., Moore Jr., T.J., Wagner, E.R.: Motorized scooter injuries in the era of scooter-shares: a review of the national electronic surveillance system. Am. J. Emerg. Med. **37**(6), 1133–1138 (2019)
4. DiMaggio, C.J., Bukur, M., Wall, S.P., Frangos, S.G., Wen, A.Y.: Injuries associated with electric-powered bikes and scooters: analysis of US consumer product data. Injury Prev. **26**(6), 524–528 (2019)
5. Che, M., Lum, K.M., Wong, Y.D.: Users’ attitudes on electric scooter riding speed on shared footpath: a virtual reality study. Int. J. Sustain. Transp. **15**, 1–10 (2020)
6. Trivedi, T.K., et al.: Injuries associated with standing electric scooter use. JAMA Netw. Open **2**(1), e187381–e187381 (2019)
7. Mitchell, G., Tsao, H., Randell, T., Marks, J., Mackay, P.: Impact of electric scooters to a tertiary emergency department: 8-week review after implementation of a scooter share scheme. Emerg. Med. Australas. **31**(6), 930–934 (2019)
8. Mayhew, L.J., Bergin, C.: Impact of e-scooter injuries on emergency department imaging. J. Med. Imaging Radiat. Oncol. **63**(4), 461–466 (2019)
9. Terrell, G.C.: Characterizing electric scooter riding behavior to detect unsafe and non-compliant use (2019)
10. Tang, C., Feng, Y., Yang, X., Zheng, C., Zhou, Y.: The object detection based on deep learning. In: 4th International Conference on Information Science and Control Engineering (ICISCE), pp. 723–728. IEEE, China (2017)
11. Chen, Z., Khemmar, R., Decoux, B., Atahouet, A., Ertaud, J.Y.: Real time object detection, tracking, and distance and motion estimation based on deep learning: application to smart mobility. In: Eighth International Conference on Emerging Security Technologies (EST), pp. 1–6. IEEE, UK (2019)

12. Kozakaya, T., Shibata, T., Yuasa, M., Yamaguchi, O.: Facial feature localization using weighted vector concentration approach. Image Vision Comput. **28**(5), 772–780 (2010)
13. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2005), vol. 1, pp. 886–893. IEEE, USA (2005)
14. Zhang, G., Huang, X., Li, S.Z., Wang, Y., Wu, X.: Boosting local binary pattern (LBP)-based face recognition. In: Li, S.Z., Lai, J., Tan, T., Feng, G., Wang, Y. (eds.) SINOBIOMETRICS 2004. LNCS, vol. 3338, pp. 179–186. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-30548-4_21
15. Leistner, C., Roth, P.M., Grabner, H., Bischof, H., Starzacher, A., Rinner, B.: Visual on-line learning in distributed camera networks. In: 2008 Second ACM/IEEE International Conference on Distributed Smart Cameras, pp. 1–10. IEEE, USA (2008)
16. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature **521**(7553), 436–444 (2015)
17. Dhande, M.: What is the difference between AI, machine learning and deep learning. Geospatial World (2017)
18. Girshick, R., Donahue, J., Darrell, T., Malik, J.: 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. IEEE, USA (2014)
19. Girshick, R.: Fast R-CNN. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1440–1448. IEEE, Chile (2015)
20. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems, pp. 91–99 (2015)
21. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: unified, real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 779–788. IEEE, USA (2016)
22. Wang, H., Zhang, Z.: Text detection algorithm based on improved YOLOv3. In: 9th International Conference on Electronics Information and Emergency Communication (ICEIEC), pp. 147–150. IEEE, China (2019)
23. Wu, F., Jin, G., Gao, M., Zhiwei, H.E., Yang, Y.: Helmet detection based on improved yolo v3 deep model. In: 16th International Conference on Networking. Sensing and Control (ICNSC), pp. 363–368. IEEE, Canada (2019)
24. Yajai, A., Rodtook, A., Chinnasarn, K., Rasmequan, S.: Fall detection using directional bounding box. In: 12th International Joint Conference on Computer Science and Software Engineering (JCSSE), pp. 52–57. IEEE, USA (2015)
25. Nowozin, S.: Optimal decisions from probabilistic models: the intersection-over-union case. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 548–555. IEEE, USA (2014)
26. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. Int. J. Comput. Vision **88**(2), 303–338 (2010)
27. Hosang, J., Benenson, R., Schiele, B.: Learning non-maximum suppression. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4507–4515. IEEE, USA (2017)
28. Yajai, A., Rodtook, A., Chinnasarn, K., Rasmequan, S.: Fall detection using directional bounding box. In: 2015 12th International Joint Conference on Computer Science and Software Engineering (JCSSE), pp. 52–57. IEEE, July 2015