INTERNATIONAL JOURNAL OF ACADEMIC RESEARCH IN BUSINESS & SOCIAL SCIENCES

Vol 11, Issue 12, (2021) E-ISSN: 2222-6990

Comparing MobileNet-SSD and YOLO v3 Learning Architecture for Real-time Driver's Fatigue Detection

Nursuriati Jamil[1], Mohammad Haziq Mohd Fadhil[2], Raseeda Hamzah[3], Muhammad Izzad Ramli[4]

[1,3,4]Faculty of Computer and Mathematical Sciences, Universiti Teknologi MARA, 40450 Shah Alam, Malaysia, [2]Mandrill Tech Sdn Bhd, A-18-5, The Vertical Business Suite, No 8, Jalan Kerinchi, Bangsar South,59200 Kuala Lumpur, Malaysia

To Link this Article: http://dx.doi.org/10.6007/IJARBSS/v11-i12/11984 DOI:10.6007/IJARBSS/v11-i12/11984

Published Date: 26 December 2021

Abstract

Convolutional Neural Network is known to achieve high accuracy in solving classification, recognition, and detection problems. In a real-time environment, time is an important factor of consideration. Even though most CNN-based architectures achieved considerably high accuracy, they are still slow even with high-end hardware. Therefore, this paper compares the time-accuracy tradeoff between two recent CNN-based learning architectures in detecting a driver's fatigue status. In our work, we define fatigue based on the rate of eye blinking. We developed a proof of concept systems, and evaluate the systems based on accuracy and detection speed. The accuracy and speed of both learning architectures were trained and tested using the Closed Eyes in the Wild (CEW) containing 1,193 closed eyes images and 1,232 opened eyes images. As MobileNet-SSD and YOLO v3 were pre-trained using a general COCO dataset, they were further configured and fine-tuned to optimize the results based on the CEW datasets, The results showed that YOLO v3 has slightly higher meanAveragePrecision(mAP) than MobileNet-SSD but slower detection speed(ms), while MobileNet-SSD proved that it has much faster speed but still maintaining high accuracy. The results of the research also showed that there is a trade-off between speed and accuracy which there was a loss of accuracy to obtain faster speed. This research also proved that lightweight MobileNet-SSD can minimize the accuracy loss to gain speed. The accuracy of the MobileNet-SSD learning model was still considered high and the detection speed was far higher than YOLO v3 learning model. Therefore, MobileNetSSD learning model was selected to have the best speed and accuracy trade-off in this research.

Keywords: Real-time Fatigue Detection, Convolutional Neural Network, MobileNet-SSD, YOLOv3

Introduction

Thirty-one percent of road accidents worldwide are caused by the driver's fatigue and drowsiness (Bhardwaj & Balasubramanian, 2018). Therefore, several measures were taken by car manufacturers to alleviate this crucial problem by using various technologies. One of it is using a driver's fatigue monitoring system to detect the onset of drowsiness in drivers, while the vehicle is in motion. There are three categories of the fatigue detection system which are based on vehicle status, behavioural, and physiological (Shakeel *et al*., 2019). The vehicle status-based detection system is a system that functions by inspecting or monitoring the car’s lane switching, steering motion, speed, pressure on the pedal and others. On the other hand, the behavioural-based system depends on the images and videos from a camera that monitors the driver's behaviour (Shamsuddin *et al.,* 2017). Finally, the physiological-based systems monitor the physiological signal emitted by the driver such as Electromyography (EMG) (Abbood *et al.,* 2014), Electroencephalography (EEG) (Khushaba *et al*., 2011), Electrocardiography (ECG) (Balasubramanian & Bhardwaj, 2019), and Electrooculography (EOG) (Bharadwaj *et al*., 2018). The vehicle status and physiological-based systems are considered not robust as the vehicle status-based is constrained to the driver's varying habits and the physiological-based systems caused annoyance and irritation to the driver as the electrodes are body intrusive (Shakeel *et al*., 2019). In this paper, the behavioural-based approach is chosen because human behaviours related to fatigues are generally consistent across culture and races. Thus, these behaviours can be taught using machine learning techniques coupled with computer vision to produce robust systems (Shakeel *et al.,* 2019).

Machine learning techniques, particularly deep learning (Saufi *et al.,* 2018) (Yahya *et al*., 2020) (Sharifuddin *et al.,* 2019) has been proven to solve real-world applications related to computer vision. Despite achieving high accuracy in classification, detection and recognition applications, the deep learning approach is still slow in execution time even with high-end hardware. Based on Liu *et al* (2016), while most deep learning-based detection systems are accurate, they are too slow to be applied in real-time. With the rapid development of deep learning, lightweight architectures (Muhammad *et al*., 2019) were introduced to solve the real-time problem. Therefore, this paper aims to investigate the recent lightweight deep learning works that have solved object detection problems in real-time. Then, we further developed a proof of concept for a fatigue detection system and evaluate the deep learning models based on the accuracy and speed. The robustness of the driver fatigue detection system does not only depend on how accurate the system is but also the time complexity needs to be low so that it can be implemented in real-time.

Related Work

In behavioural-based fatigue detection, the driver's eye status is obtained using video images in real-time. The process begins with face detection followed by eyes localization. Once the eyes are localized, the driver's fatigue status is determined using the theory of PERCLOS (Percentage of Eyelid Closure Over the Pupil Time) introduced by (Wierwille *et al.,* 1994). The metric PERCLOS is calculated by counting the number of frames in which there was no pupil detected and dividing this by the total number of frames for a specific time interval. The higher the number of PERCLOS, the higher the driver's fatigue status. In 2017, Chen *et al.,* the face detection was done using the skin colour feature, and the horizontal and vertical curve pitch of the eyes were used to localize the exact position of the eyes. The eye state was determined by using the percentage of the pupil opening. If the pupil is 20% opened, the eye is considered as opened, and closed, otherwise. The fatigue status was measured using PERCLOS. The YCbCr colour space used in this study can better separate the brightness component from the colour image to reduce the impact caused by light interference. However, the eyes cannot be located if they were occluded, thus significantly reduced the detection effectiveness. In their work, Sravan *et al* (2018) used the Viola-Jones algorithm for face detection and the eyes localization. Simple image processing techniques were then employed to convert the eyes into binary images, and the percentage of white and black was calculated using a threshold value to determine whether the eyes were opened or closed. The fatigue status was detected using PERCLOS. This method was simple, did not require prior information of the image, computationally inexpensive and was practical for real-time implementation. However, the thresholding technique was not robust to illumination and colour variance, and it was also overly sensitive to noise.

Recent work of fatigue detection showed an increase in using Convolutional Neural Network (CNN) to overcome the limitations of previous work. The research done by Shakeel *et al.* (2019) used a Convolutional Neural Network (CNN) architecture known as MobileNet together with Single Shot Detector (SSD). The MobileNet-SSD platform was trained using different datasets: 1) FDDB (Jain & Miller, 2010) comprising 5,171 faces in 2,845 images 2) YawDD (Abtahi *et al*., 2014) consisting of two sets of videos that were recorded using two different camera locations inside a car 3) 1,192 images from Closed Eyes in the Wild (CEW) dataset and 4) 2,691 custom images (Song *et al*., 2014 It was then used to detect the face, open and closed eye with an average accuracy of 83.7%. The proposed MobileNet-SSD performed in a wide variety of conditions, with varying illumination conditions, poses and occlusions, and in real-world driving. However, due to the lack of training in low lightings, the accuracy reduced when the lighting came from the backside of the camera lens. Another notable work of fatigue detection was done by Jabbar *et al*. (2020) using Faster Region Convolutional Neural Network (Faster R-CNN). The model was trained using National Tsing Hua University (NTHU) Driver Drowsiness Detection Dataset (Weng *et al*., 2016) which included 22 different ethnicities of subjects in day and night conditions. This research produced a relatively high accuracy of 83.3% to detect facial landmarks. However, occlusion and bad lighting conditions affected the accuracy significantly.

To fulfill the aim of this study, a literature search on previous work of object detection using CNN-based models was conducted. In Huang *et al* (2017), a speed/accuracy trade-off evaluation was done between Faster Region-CNN (Faster R-CNN) (Ren *et al*., 2015), Region Fully Connected Network (R-FCN) (Dai *et al*., 2016) and Single Shot Detection (SSD) (Liu *et. al.*, 2016). Six different feature extractors and other critical parameters were used to achieve fair comparisons. The training pipelines for SSD, Faster R-CNN and R-FCN were recreated in TensorFlow as the detection platform. The results showed that R-FCN and SSD models were faster on average, compared to Faster R-CNN. However, Faster R-CNN produced more accurate models. Out of all the fastest models, SSD models with Inception v2 and MobileNet feature extractors are the most accurate. Even though SSD models typically performed poorly on small objects, they are competitive with Faster RCNN and R-FCN on large objects, even outperforming the faster and more lightweight feature extractors. As with speed, SSD-MobileNet was the cheapest, requiring less than 1Gb (total) memory in almost all settings. A more recent work of (Redmon & Farhadi, 2018) compared variations of You Look Only Once (YOLO), RetinaNet and SSD learning models using COCO datasets. YOLO v3 showed the fastest detection speed of 22ms and achieved second place for mean Average Precision (mAP) of 55.3 after FPN-FRCN at 59.1. Even though the FPN_FRCN achieved the highest accuracy, the time taken was 172ms. Therefore, considering the time and accuracy trade-off, we chose to implement fatigue detection using YOLO v3 and MobileNet-SSD.

Methodology

In this work, the CNN architectures MobileNet-SSD from Shakeel *et al*. (2019) and YOLO v3 from Redmon *et al.* (2018) were adapted based on literature where MobileNet-SSD showed great accuracy compared to traditional method while YOLO v3 showed great accuracy and detection time. To compare both deep learning architectures, they were trained and evaluated using the same dataset. The research process flowchart of this research is shown in Fig 1. The dataset used was the Closed Eyes in the Wild (CEW) containing 1,193 closed eyes images and 1,232 opened eyes images (Song *et al*., 2014). The images resolution is 100 × 100 pixels comprising a variety of people of different ethnicity, various skin colours, different ages, different facial features, different conditions of lighting and people with glasses and without glasses. See Fig. 2. For the dataset pre-processing, the images were annotated and labelled using *LabelImg* image annotator into two classes (i.e. Opened and Closed). Annotating the images involved drawing the bounding boxes and the labelling is the naming of the classes of the bounding boxes. Fig. 3 shows the *LabelImg* with bounding boxes drawn onto the images. The annotations were later saved in .xml format for the MobileNet-SSD and were converted in .txt format for the YOLO v3. Eighty percent (1,963 images) of the dataset from each class was further used for training and twenty percent of dataset from each class was used for testing. The next step was to develop the learning architectures and configured them for training and testing.

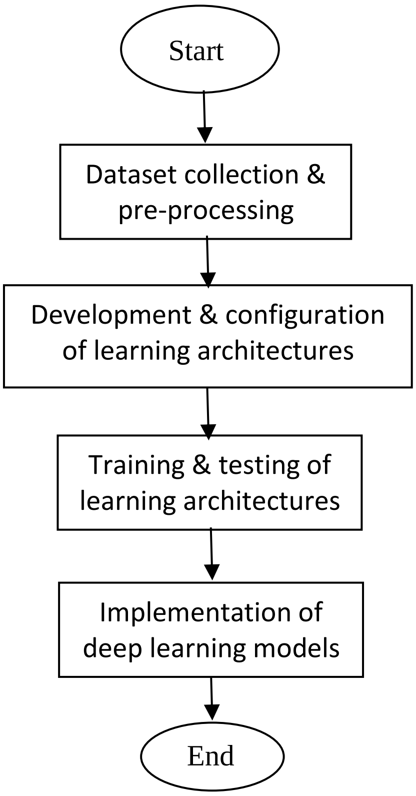

Start
Dataset collection & pre-processing
Development & configuration of learning architectures
Training & testing of learning architectures
Implementation of deep learning models
End

Fig 1.: Experiments flowchart

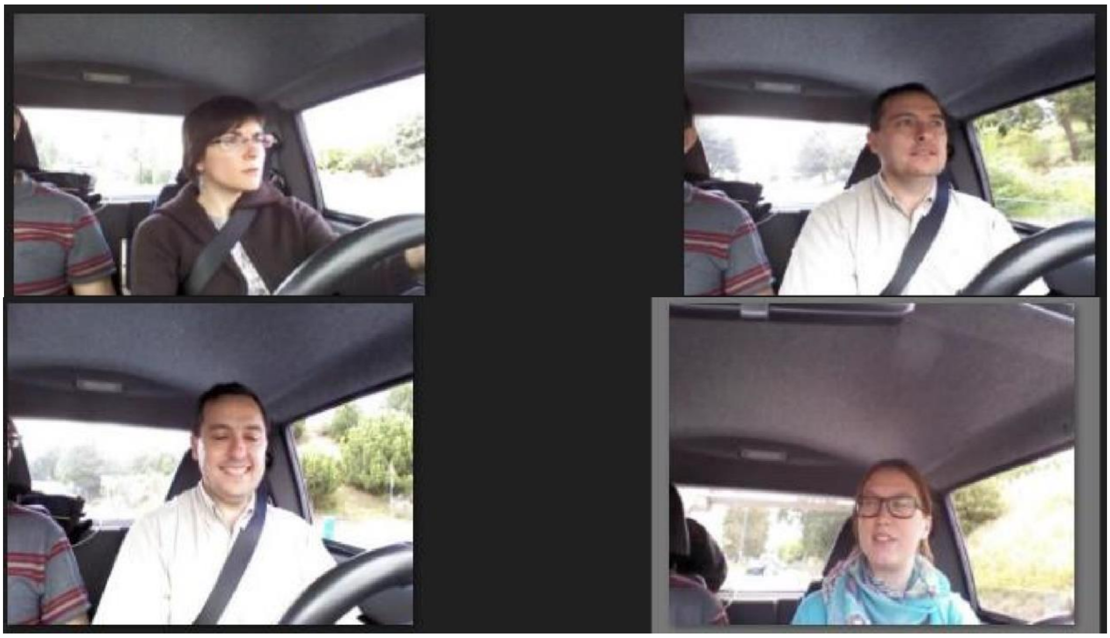
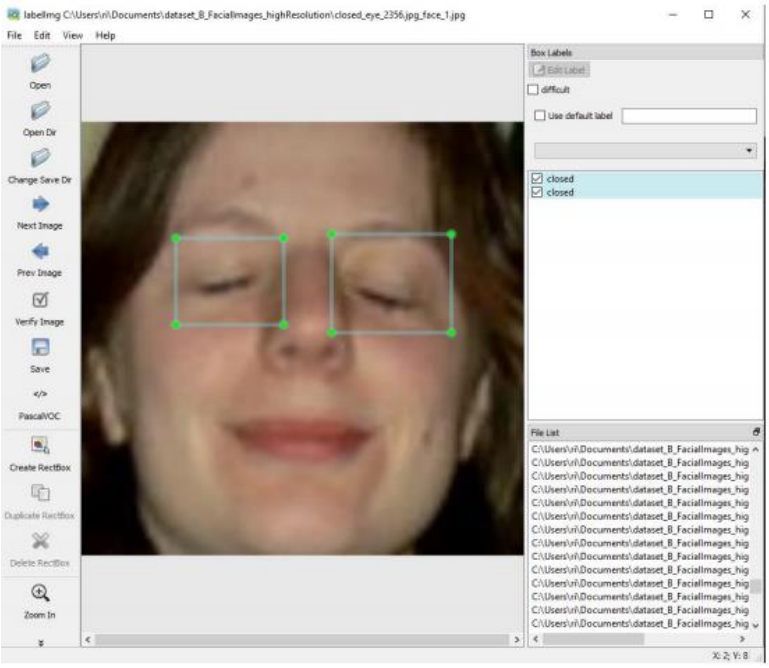


Fig. 2: Samples from CEW image dataset (Top: Opened eyes, Bottom: Closed eyes)



labelImg C:\Users\ri\Documents\dataset_B_FacialImages_highResolution\closed_eye_2356.jpg_face_1.jpg

☐ difficult

☐ Use default label

☑ closed
☑ closed

File List

```
C:\Users\ri\Documents\dataset_B_FacialImages_hig
C:\Users\ri\Documents\dataset_B_FacialImages_hig
C:\Users\ri\Documents\dataset_B_FacialImages_hig
C:\Users\ri\Documents\dataset_B_FacialImages_hig
C:\Users\ri\Documents\dataset_B_FacialImages_hig
C:\Users\ri\Documents\dataset_B_FacialImages_hig
C:\Users\ri\Documents\dataset_B_FacialImages_hig
C:\Users\ri\Documents\dataset_B_FacialImages_hig
C:\Users\ri\Documents\dataset_B_FacialImages_hig
C:\Users\ri\Documents\dataset_B_FacialImages_hig
C:\Users\ri\Documents\dataset_B_FacialImages_hig
C:\Users\ri\Documents\dataset_B_FacialImages_hig
C:\Users\ri\Documents\dataset_B_FacialImages_hig
C:\Users\ri\Documents\dataset_B_FacialImages_hig
```

X: 2; Y: 8

Fig 3: *LabelImg* with bounding boxes (Class: Closed)

The development of learning architectures involves the installation of software tools and libraries. For MobileNet-SSD, the Tensorflow Object Detection API repository from GitHub was cloned into the local machine and configured. Meanwhile, for YOLO v3 the Darknet repository (Redmon, 2016) was cloned into the cloud machine which is the Google Colab and was configured. Yolo V3 configurations were fine-tuned as follows:

batch=64; subdivisions=16; max_batches = 4000; steps=3200, 3600

The ‘batch’ was the size of the dataset loaded per iteration. The ‘subdivisions’ was the fraction of ‘batch’ that was sent to the GPU for processing until the total ‘subdivisions’ sent to GPU was equal to the number of ‘batch’. The ‘max_batches’ was the maximum number of iterations for the training. The ‘steps’ was the number of iterations where the learning rate was reduced ten times from the previous learning rate. To reduce GPU memory overload, the ‘batch’ was set to 64 and ‘subdivisions’ was set to 16. The filter of every convolutional layer was edited to reduce time complexity based on its number of class. We used 21 filters as calculated from (1) (Redmon, 2016).

$$\mathrm{N}umber\ of\ filters = 3(number\ of\ classes + 5) \tag{1}$$

The training phase was done when the image dataset together with annotation files were fed into the learning architecture. For the training, we used the pre-trained YOLO v3 and MobileNet-SSD that were trained using Microsoft Common Objects in Context (COCO) dataset (Lin *et al*., 2014). The dataset contains 328,000 images, 2.5 million labeled instances and 91 object types. The training of MobileNet-SSD was done on a computer with Intel Core i5-7300HQ 4-cores CPU that can handle 4 threads, and maximum clock speed of 3.5GHz. with a GPU of Nvidia GTX 1050 that has Video RAM(VRAM) of 4GB. The RAM used in this paper contained 8GB of memory with speed of 2400 MH. Training on YOLO v3 used the Google Colab platform which provided CPU of Intel Xeon 2.3GHz dual core processor and GPU of Nvidia Tesla K80 which consists of 12GB VRAM. Google Colab also provides 13GB of RAM with 60GB of storage. During training, the predicted bounding boxes and classes were compared with the ground truth object to obtain the mean Average Precision (mAP). The prediction time for each image also was recorded to obtain the average time (ms) so that the trade-off between speed and accuracy of the two models can be analysed. The MobileNet-SSD model started training every 5 minutes, the model saved the checkpoint of the training and automatically ran the testing. For YOLO v3 in Google Colab, the weight of the training was saved for every 100 steps.

To determine if the driver is drowsy or not, the threshold value for video is required for the duration of eyes closed and the eyes blinking rate. The blinking of eyes was stated to take a duration of 0.4s (Shakeel *et al*., 2019). In a 30 frames per sec (fps) video, the threshold value was calculated as in (2).

$$\text{Threshold} = 0.4 \text{ x fps}(30) \tag{2}$$

A normal person blinked more than ten times per minute, while a sleepy person blinked less (Pasaribu *et al.,* 2019). If a person closed and opened his/her eyes in the duration of 0.4 second, the blink counter increased. If the blink rate was below ten times per minute, then the driver was considered fatigue/drowsy. For the duration of eyes closure, if the driver’s eyes were closed more than 0.5 seconds, the driver was considered as fatigue/drowsy (Islam, 2019).

Evaluations of the two learning models were done using accuracy measured by meanAveragePrecision (mAP) and detection speed (ms). Calculation of mAP involved the computation Average Precision (AP) for each class, in our case Opened and Closed classes. The understanding of AP requires the understanding of intersection over union (IoU), which is the ratio of area of intersection and the area of union between the predicted and ground truth bounding boxes (Ren, 2019). IoU is responsible to determine whether the predicted bounding box is True Positive (TP), False Positive (FP) or False Negative (FN). Typically, the prediction is considered as TP if the IoU is more than 0.5. The bounding box is considered as

FP if the IoU is less than 0.5. If the predicted bounding box has IoU more than 0.5 but detected as a wrong class, it is considered as FN. AP is defined as the area under Precision-Recall (PR) curve. Recall can be simply described as the True Positive (TP) rate of the detected objects and Precision measures how accurate is your predictions. Before PR curve is plotted, the Interpolated Precision, *Pinterp*, was calculated at each recall level, *r*, by taking the maximum precision measured for thar *r* as seen in (3).

$$p_{interp}(r) = \ max_{r'\geq r}^{p(r')} \qquad (3)$$

AP per class was then computed by summing IP at 11 different levels starting from 0 until 1. The calculation of AP can be seen in (4).

$$AP \ = \frac{1}{11}(APr(0) + Apr(0.1) + \cdots + Apr(1.0)) \qquad (4)$$

Finally, mAP was calculated by taking the AP of each class.

Results and Discussion

In this section the accuracy (mAP) and detection speed (ms) of MobileNet-SSD is presented followed by YOLO v3. Finally, comparison of the two learning models was done. The training of MobileNet-SSD took approximately five hours and 128,000 steps. The overall mAP scored at 53.29% with average loss of 2.967. Fig. 4 shows the mAP of classes averaged over IOU thresholds ranging from 0.5 to 0.95 with 0.05 increments. Based on Fig. 4, the gradient of the training decreased after 1 hour of training at 25,000 steps. Then the mAP slightly increased until the end of the training.

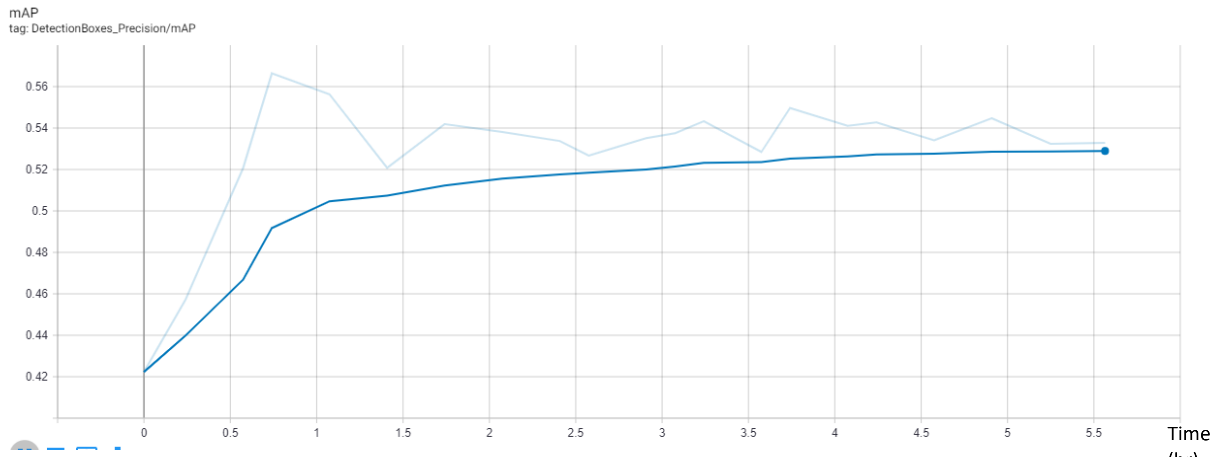


Fig. 4: Overall mAP of MobileNet-SSD

In MobileNet-SSD, the mAPs of detected objects were evaluated in different sizes. Large objects were defined as objects with sizes more than 96 × 96 pixels, medium objects were objects with sizes ranging from 32 × 32 pixels to 96 × 96 pixels, mAP of small objects were for objects with sizes below 32 × 32 pixels. The mAP of large objects remained constant at -1 as all the eyes were smaller than 96 × 96 pixels, thus the mAP was deemed irrelevant. For medium-sized objects, the mAP of MobileNet-SSd achieved 58.33%, while the mAP of small-sized objects scored lower at 53.34%. This is aligned with Tsang (2018) stating that

MobileNet-SSD has difficulties in detecting small objects. To find the optimal threshold for MobileNet-SSD, Optimal Threshold Tuning was done by evaluating the mAP at thresholds 0.5 and 0.75 IOU. The mAP at 0.5 IoU was 97.8% while the mAP at 0.75 IoU was 51.28%. This shows that the better threshold value was at IoU 0.5. The detection speed of the MobileNet-SSD was 135.5ms running on laptop with Nvidia GeForce 940MX GPU. For YOLO v3 learning model, the Darknet platform can only run evaluation using threshold of 0.5 IOU. The mAP of the YOLO v3 at 0.5 IOU was 99.14% which was slightly better than the MobileNet-SSD. The detection speed of YOLO v3 that was run on laptop with Nvidia GeForce 940MX was 497.51ms.

The results were summarized in Table 1. For comparing MobileNetSSD and YOLO v3, the mAP at threshold of 0.5 IOU was chosen because the optimal mAP on MobileNet-SSD was at 0.5 IOU and YOLO v3 was only able to be tested on mAP at 0.5 IOU threshold. Based on Table 1, YOLO v3 showed a slightly better mAP at 0.5 IOU which made it slightly better at the eye detection and classification. On the other hand, MobileNet-SSD showed a faster detection than YOLO v3 which was 3.67 times faster. To find the best trade-off between detection speed and accuracy, the difference between both detection time and accuracy (mAP) was determined. Based on Table 1, YOLO v3 showed 1.34% higher mAP than MobileNetSSD but extremely slower detection speed by a factor of 3.67. The results are in contrast with the research made by Redmon (2018) that mentioned that YOLO v3 can detect three times faster than the SSD model but the results of this research showed that the deep learning model that showed faster detection speed was the MobileNet-SSD.

Table 1:
Comparisons of MobileNet-SSD and YOLO v3 Accuracy and Detection Speed

Result\Model	MobileNet-SSD	YOLO v3
mAP Overall	53.29%	NA
mAP (Large Object)	-1	NA
mAP (Medium Object)	58.33%	NA
mAP (Small Object)	53.34%	NA
mAP(0.5IOU)	97.8%	99.14%
mAP(0.75IOU)	51.28%	NA
Detection Speed(ms)	135.5	497.51

The detection speed and accuracy from the work done by Shakeel *et al*. (2019) to construct the MobileNet-SSD model in this research was 200ms and the mAP at 0.5 IOU was 83.7%. The results of this research showed that the MobileNet-SSD deep learning model constructed in this research was better in terms of accuracy and speed than the MobileNet-SSD in the work done by Shakeel et al. (2019) for driver's fatigue detection. The detection speed for this research also was faster than the work done by Shakeel et al. (2019) because this research used a better hardware to make predictions at the frames of webcam's videos. Shakeel et al. (2019) only used Raspberry Pi 3 and mobile devices to make predictions on webcam video and this resulted in slower detection speed. As for the YOLO v3 that adopted Redmon (2018)'s work, the results were only better for the mAP which was 99.14% while work from Redmon (2018) achieved only 57.3%. The detection speed of YOLO v3 in this research was much slower which was 497.51ms in average while the YOLO v3 from Redmon (2018)'s achieved the average detection speed of 22ms. The detection speed for YOLO v3 and

MobileNet-SSD in this research was slower than the detection speed of YOLO v3 and SSD in research conducted by Redmon (2018). This is because the hardware used by Redmon (2018) was Nvidia Titan X that was significantly better than hardware used in our work that only used Nvidia 940MX to run the detection on webcam videos.

Conclusion

The research of driver's fatigue detection is crucial for future integration of deep learning models into the real-time systems. Two deep learning models which are MobileNet-SSD and YOLO v3 were compared in terms of accuracy and detection speed of a driver's fatigue. Even though, MobileNet-SSD scored a slightly lower mean Average Precision from YOLO v3, but the detection speed of MobileNet-SSD surpassed YOLO v3 by almost quadrupled. Therefore, we concluded that MobileNet-SSD has a better speed and accuracy tradeoff compared to YOLO v3. However, further work needs to be done to improve the tradeoff. The limitation of the research is that the training and testing datasets used only images with resolution of 100 × 100 pixels. This caused the training to miss some features in the closed and opened eyes. Besides that, this research also did not cover the detection during nighttime because the dataset did not include the images during night time such as infrared images captured from infrared cameras. Detection speed can be improved by removing the unnecessary neuron in the CNN-based models and this method is called pruning. Pruning reduces the complexity of a network to remove unwanted computations and subsequently increases the detection speed.

Acknowledgements

Due acknowledgement is accorded to the Research Management Centre, Universiti Teknologi MARA, Shah Alam, Selangor for the funding received for this publication through LESTARI grant scheme no. 600-RMC/LESTARI SDG-T 5/3 (139/2019).

References

Abbood, H., Al-Nuaimy, W., Al-Ataby, A., Salem, S. A., & AlZubi, H. S. (2014). Prediction of driver fatigue: Approaches and open challenges. In *2014 IEEE 14th UK Workshop on Computational Intelligence (UKCI)* (pp. 1-6).

Abtahi, S., Omidyeganeh, M., Shirmohammadi, S., & Hariri, B. (2014). YawDD: A yawning detection dataset. In *Proceedings of the 5th ACM Multimedia Systems Conference* (pp. 24-28).

Balasubramanian, V., & Bhardwaj, R. (2018). Grip and electrophysiological sensor-based estimation of muscle fatigue while holding steering wheel in different positions. *IEEE Sensors Journal*, *19*(5), 1951-1960.

Bharadwaj, S., & Kumari, B. (2017). Electrooculography: Analysis on device control by signal processing. *International Journal of Advanced Research in Computer Science*, *8*(3).

Bhardwaj, R., Natrajan, P., & Balasubramanian, V. (2018). Study to Determine the Effectiveness of Deep Learning Classifiers for ECG Based Driver Fatigue Classification. In *2018 IEEE 13th International Conference on Industrial and Information Systems (ICIIS)* (pp. 98-102).

Chen, P. (2017, October). Research on driver fatigue detection strategy based on human eye state. In *2017 IEEE Chinese Automation Congress (CAC)* (pp. 619-623).

Dai, K. J., & R-FCN, Y. L. (2016). Object detection via region-based fully convolutional networks. arxiv preprint. In *arXiv preprint*.

Huang, J., Rathod, V., Sun, C., Zhu, M., Korattikara, A., Fathi, A., ... & Murphy, K. (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).

Islam, A., Rahaman, N., & Ahad, M. A. R. (2019). A Study on Tiredness Assessment by Using Eye Blink Detection. *Jurnal Kejuruteraan*, *31*(2), 209-214.

Jabbar, R., Shinoy, M., Kharbeche, M., Al-Khalifa, K., Krichen, M., & Barkaoui, K. (2020, February). Driver drowsiness detection model using convolutional neural networks techniques for android application. In *2020 IEEE International Conference on Informatics, IoT, and Enabling Technologies (ICIoT)* (pp. 237-242).

Jain, V., & Learned-Miller, E. (2015). Face detection data set and benchmark. *URL http://vis-www.cs.umass.edu/fddb/results.html*.

Lin, T., Maire, M., Belongie, S. J., Bourdev, L. D., Girshick, R. B., Hays, J., ... & Zitnick, C. L. (2014). Microsoft COCO: common objects in context. CoRR abs/1405.0312 (2014). *arXiv preprint arXiv:1405.0312*.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016). Ssd: Single shot multibox detector. In *European Conference on Computer Vision* (pp. 21-37). Springer, Cham.

Khushaba, R. N., Kodagoda, S., Lal, S., & Dissanayake, G. (2010). Driver drowsiness classification using fuzzy wavelet-packet-based feature-extraction algorithm. *IEEE Transactions on Biomedical Engineering*, *58*(1), 121-131.

Muhammad, N. A., Ab Nasir, A., Ibrahim, Z., & Sabri, N. (2018). Evaluation of CNN, Alexnet and GoogleNet for fruit recognition. *Indonesian Journal of Electrical Engineering and Computer Science*, *12*(2), 468-475.

Sharifuddin, M. S. I., Nordin, S., & Ali, A. M. (2019). Voice Control Intelligent Wheelchair Movement Using CNNs. In *2019 1st International Conference on Artificial Intelligence and Data Sciences (AiDAS)* (pp. 40-43). IEEE.

Pasaribu, N. T. B., Prijono, A., Ratnadewi, R., Adhie, R. P., & Felix, J. (2019). Drowsiness detection according to the number of blinking eyes specified from eye aspect ratio value modification. In *1st International Conference on Life, Innovation, Change and Knowledge (ICLICK 2018)*. Atlantis Press.

Redmon, J. (2016). Darknet: Open source neural networks in C (2013–2016). *URL: https://pjreddie.com/darknet/*

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

Ren, S., He, K., Girshick, R., & 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).

Saufi, M. M., Zamanhuri, M. A., Mohammad, N., Ibrahim, Z. (2018). Deep Learning for Roman Handwritten Character Recognition. *International Journal of Electrical Engineering and Computer Science*, *12*(2), pp.455-460.

Shakeel, M. F., Bajwa, N. A., Anwaar, A. M., Sohail, A., & Khan, A. (2019). Detecting driver drowsiness in real time through deep learning based object detection. In *International Work-Conference on Artificial Neural Networks* (pp. 283-296). Springer, Cham.

Shamsuddin, M. R. B., Sahar, N. N. B. S., & Rahmat, M. H. B. (2017, November). Eye Detection for Drowsy Driver Using Artificial Neural Network. In *International Conference on Soft Computing in Data Science* (pp. 116-125). Springer, Singapore.

Song, F., Tan, X., Liu, X., & Chen, S. (2014). Eyes closeness detection from still images with multi-scale histograms of principal oriented gradients. *Pattern Recognition*, *47*(9), 2825-2838.

Sravan, C., Onesim, K. J., Bhavana, V. S. S., Arthi, R., & Srinadh, G. (2018). Eye Fatigue Detection System. In *2018 International Conference on System Modeling & Advancement in Research Trends (SMART)* (pp. 245-247).

Tsang, S.-H. (2018). Review: SSD — Single Shot Detector (Object Detection). *URL: https://towardsdatascience.com/review-ssd-single-shot-detector-object- detection 851a9460 -7d11*

Weng, C. H., Lai, Y. H., & Lai, S. H. (2016). Driver drowsiness detection via a hierarchical temporal deep belief network. In *Asian Conference on Computer Vision* (pp. 117-133). Springer, Cham.

Wierwille, W. W., Wreggit, S. S., Kirn, C. L., Ellsworth, L. A., & Fairbanks, R. J. (1994). *Research on vehicle-based driver status/performance monitoring; development, validation, and refinement of algorithms for detection of driver drowsiness. Final report* (No. HS-808 247).

Yahya, M. A., Abdul-Rahman, S., & Mutalib, S. (2020). Object Detection for Autonomous Vehicle with LiDAR Using Deep Learning. In *2020 IEEE 10th International Conference on System Engineering and Technology (ICSET)* (pp. 207-212).