Traffic Sign Detection Using Convolutional Neural Network on Autonomous Vehicle System

Rabbi Fijar Mayoza
School of Electrical Engineering and Informatics
Institut Teknologi Bandung
Bandung, Indonesia
fijarmayoza.rabbi@gmail.com

Dr. Ir. Rinaldi Munir, M.T.
School of Electrical Engineering and Informatics
Institut Teknologi Bandung
Bandung, Indonesia
rinaldi@informatika.org

***Abstract*—Artificial intelligence (AI) gives autonomous vehicle ability to drive by itself. It is believed that AI can reduce the risk of accidents caused by human error. One example of AI implementation on autonomous vehicle is visual system for detecting traffic sign. Convolutional Neural Network (CNN), a part of deep learning method, is used in this research to build traffic sign detection model for Indonesia. However, dataset is needed by this method to perform training. The unavailability of Indonesian traffic sign dataset may become challenge in building the model due to the distinct characteristics of traffic sign among countries. The proposed solution is to feed Extended Malaysian Traffic Sign Dataset (EMTD) into CNN to produce the detection model by reason that it contains traffic signs that are similar to Indonesian traffic signs. The solution adapts Faster R-CNN model which has been developed for detecting foreign traffic sign. The CNN model is coded with Python 3 using Keras-Tensorflow library. Input data preprocessing includes resize, histogram equalization, augmentation, and pixel scaling. This research evaluates and compares the performance of Faster R-CNN with VGG16 and ResNet50 backbone.**

Keywords—Traffic sign detection; Faster R-CNN; RPN; mean average precision; frame per second

I. Introduction

Autonomous vehicle is expected to eliminate human dependency to control the vehicle. Therefore, it will reduce tension and fatigue of driving which can prevent the driver from making mistake. Around 90% of road accidents have been caused by human error. Implementation of AI on autonomous vehicle system is claimed to be able to reduce the 90% of road accidents [1]. One example of its implementation is visual system for detecting traffic sign. The system uses camera as sensor and receives video as input data. By machine learning technique, the system has ability to identify the existence of traffic sign from video.

There are eight challenges in developing a traffic sign detection and recognition system. They are variable lightning condition, fading and blurring, limited visibility, multiple appearance at once, motion artifacts, damaged or partially obscured, unavailability of public dataset, and real-time application [2]. The previous research proposed traffic sign detection using RGB normalization and thresholding for region of interest (ROI) segmentation. This has weakness when the size of traffic signs changes as the vehicle moves. It is impossible to do labeling and calculate the labeled area manually.

CNN as a part of deep learning method is so popular in computer vision task. It is because the architecture of CNN is especially designed to handle image-based data. Many traffic sign detection and recognition research prove that CNN-based model produces a model that good at handling bad quality sign and partially obscured sign [2]. Besides, CNN runs feature extraction faster and generates a greater number of information than traditional image segmentation. It is why CNN is more reliable to be implemented for autonomous vehicle system than pure traditional image processing approach [3,4].

The unavailability of public dataset for Indonesian traffic sign detection may be an obstacle. Public dataset like German Traffic Sign Detection Benchmark (GTSDB) or Laboratory for Intelligent and Safe Automobiles (LISA) has different shapes and types of traffic sign than Indonesian traffic sign. Meanwhile, Malaysia already has had a public dataset, EMTD, which contains traffic signs that are similar to Indonesian traffic signs. Therefore, EMTD will be used as data source to build the detection model.

The proposed solution is to feed Extended Malaysian Traffic Sign Dataset (EMTD) into CNN to produce traffic sign detection model. The architecture will be Faster R-CNN which is adapted from an open source model. It was trained and tested with GTSDB. To be trained and tested with EMTD, some modification and retraining is done. Final model then will be evaluated by applying the model to a road trip video that took place in Indonesia.

II. Literature Review

A. *Traffic Sign and Dataset Review*

Traffic signs play a vital role in directing, informing and controlling road users' behavior in an effort to make the roads as safe as possible for everyone [5]. Three types of traffic sign are instruction, warning, and information. Each type has some unique characteristics that make it distinguishable. Generally, the instruction sign has a circular form, the warning sign has a triangular or rectangular form, and the information sign has rectangular or square form. But it may slightly vary in some countries.

As the alternative of Indonesian traffic sign dataset, EMTD is used because of the similarity. Fig.1. shows how similar are the Malaysian traffic signs compared to Indonesian traffic signs. EMTD consists of 763 image files in both jpg and png with various sizes. It has also an annotation text file which saves the traffic sign pixel coordinates on each image file. There are 1452 annotated traffic signs in total. As the label, the annotation will be generated as ground truth bounding box and ground truth class later in the model. The dataset is split into train, validation, and test. The ratio of train to validation is 85:15. While the test set is 20% of whole dataset.

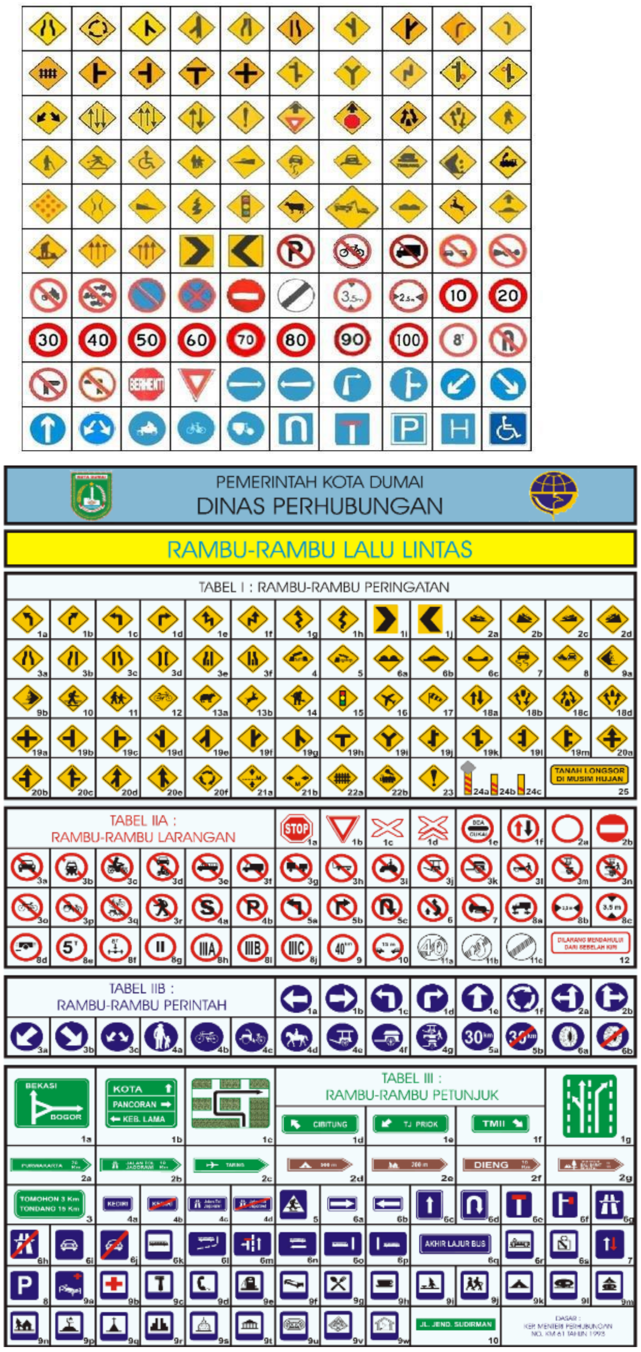
PEMERINTAH KOTA DUMAI
DINAS PERHUBUNGAN
RAMBU-RAMBU LALU LINTAS
TABEL I : RAMBU-RAMBU PERINGATAN
TABEL IIA : RAMBU-RAMBU LARANGAN
TABEL IIB : RAMBU-RAMBU PERINTAH
TABEL III : RAMBU-RAMBU PETUNJUK

Fig. 1. Malaysian traffic signs (top) compared to Indonesian traffic signs (bottom).

B. *Faster R-CNN*

The detection algorithm used in this research is Faster R-CNN. Proposed by Ren with Girshick, the author of Fast R-CNN, it improves Fast R-CNN by taking out the selective search [6]. Instead, it uses region proposal network (RPN) to find ROI. So the whole network is separated into three part which are base net / feature extractor, RPN, and classifier net /Fully Connected. RPN allows the network to determine ROI by learning, not using greedy approach like Fast R-CNN. Fig. 2 shows the architecture of Faster R-CNN.

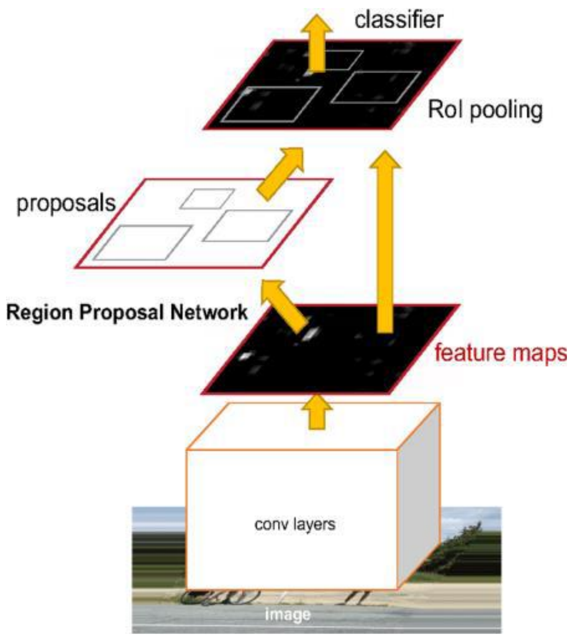

Fig. 2. Faster R-CNN Network [6]

The Faster R-CNN contains a base CNN which extracts the feature maps from the raw image. The input of the CNN is the training image and the output of the CNN is the feature map corresponding to the input image. While the previous research of traffic sign detection used VGG16 and ZFNet for experiment [4], VGG16 and ResNet50 will be implemented as the base CNN in the experiment.

The next important part of Faster R-CNN is RPN. The RPN generates different proposals which may contain the object to be detected by using four coordinates as a bounding box. Besides the proposals, the RPN also gives the information about the confidence of each proposal as binary classification scores. the input of the RPN is the feature maps generated by the base net, and the output of the RPN are positive proposals. The positive proposal is determined by several reference boxes called anchors. By comparing these anchors with the ground truth boxes, we get the positive proposals which may contain the traffic sign objects. The anchors stride across the feature maps. The positive proposals that really contains the traffic sign is a ROI.

ROI pooling is done then to standardize different anchors in the same size, since the RPN generate several proposals with different sizes. To feed the region proposals into classifier network, all ROI must be standardized into a fixed size. The adjustment of the proposals includes cropping and resizing. Lastly, there is a classifier network which contains two separate fully connected layers. Specifically, one is for classification of traffic sign type denoted as class number and confidence score, another is for regression of the box location denoted as 4 coordinates (x,y,w,h). This classifier takes the ROI pooling results as the input and generates predicted boxes and make the classification.

III. Implementation and Evaluation

The implementation of the solution is based on the open source code shared by previous research [4]. Some changes are made to accommodate the dataset and the problem and also to enhance the performance. It is written in Python3 with Keras 2.2.4 and OpenCV2. The computational resource is another important factor when doing a deep learning project. The Faster R-CNN model needs a lot of computational resources which requires GPU computing. Due to some limitations, this research only uses free version of Google Colab. It provides Intel(R) Xeon(R) @ 2.00GHz CPU. NVIDIA Tesla P100-PCIE-16GB and NVIDIA Tesla T4 GPUs are used during training. The solution is implemented by following the design which is given on Fig.3.

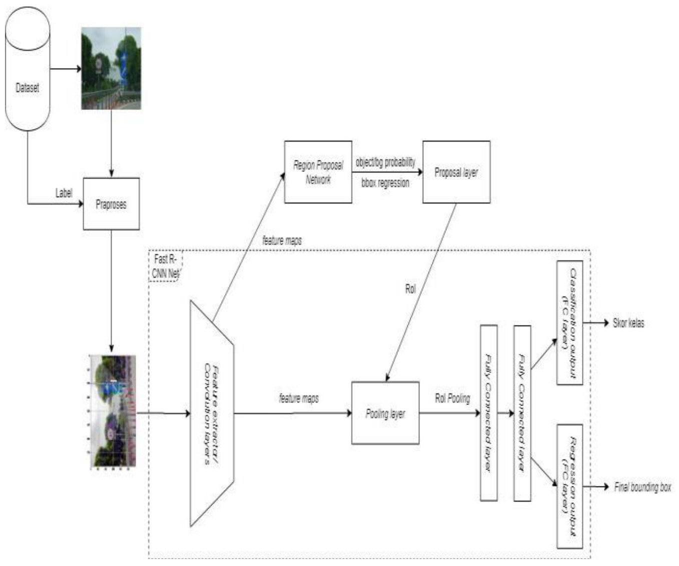

Fig. 3. Solution design

A. Data Preprocessing

Data preprocessing includes resizing, histogram equalization, augmentation, and pixel scaling. Resizing is a dimension reduction technique by changing the image size into fixed and smaller value. If length size is longer than width size then the image size becomes 600 px * (ratio of actual length size to actual width size) X 600 px. Else it becomes 600 X 600px * (ratio of actual length/ to actual width size. After resizing it, we enhance the image quality by using histogram equalization technique. Then the augmentation is done to the image by applying random rotation of 90, 180, or 270 degrees. Finally Fig.4. While the pixel value of the image is scaled by using zero-center techniques. All these data preprocessing tasks are done by data generator. Fig.4. shows the output of data generator.

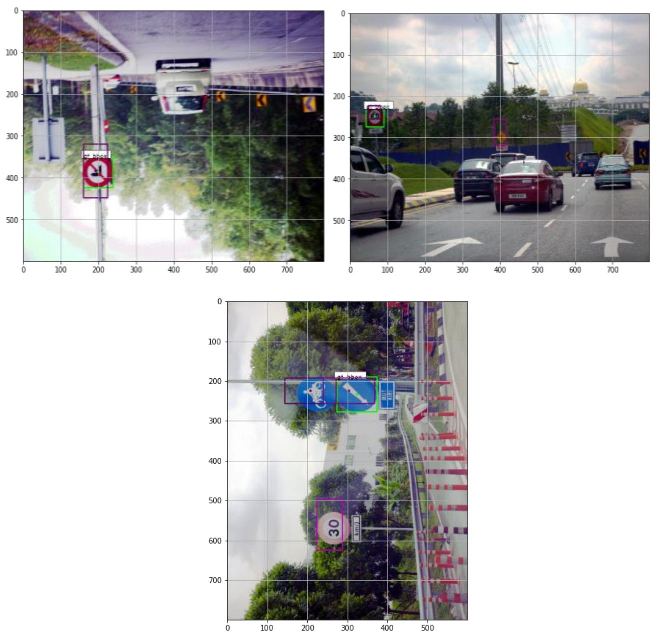

Fig. 4. The output of data generator, green box is ground truth box and purple box is anchor box

B. Training Configuration

As our traffic sign detection model is intended to be deployed on autonomous vehicle system, it aims to perform within a small amount of time with fairly high accuracy. But to reduce the problem space of this thesis, we decide to set the scope:

- The solution will be tested on a non-real-time video.
- The solution is assumed not to encounter bad weather condition.
- The solution's configuration is set for traffic sign detection in Indonesia.

There are two types of configuration for training our model. The first one is *default configuration* which uses same configuration with Nie's [4]. The second one is *best modification configuration* which chooses the best config of several attempts from experiment. Default configuration uses VGG16 as feature extractor, while the modification uses ResNet50. The anchor box scales for both configurations are 8, 16, 32, and 64 pixels. The anchor box ratios for both configurations are 1:1, 1:2, and 2:1. The RPN stride has different value for each configuration, 8 pixels for default and 16 pixels for modification. It is because it depends on the architecture of the CNN.

The training is done in two steps in order to speed up the training time. For the first step, we train RPN only. In the next step, we train whole Faster R-CNN by loading the RPN weight obtained from the previous step. The maximum epoch number for RPN training is 20, while for the whole network training is 30. There 500 batches during one epoch of training. The loss function used in our network is defined by Ren [4] as

$$cross\_entropy(predicted\_class, actual\_class) \quad (1)$$

for classification loss and

$$l_u = \sum_{i\in x,y,w,h} smooth_{L1}(u_i(predicted) - u_i(target)) \quad (2)$$

for bounding box loss with the smooth function as

$$smooth_{L1}(x) = \begin{cases} x^2/2, & \|x\| < 1 \\ \|x\| - 0.5, & otherwise \end{cases} \quad (3)$$

The detections are considered to be true if the predicted bounding box overlaps with at least 50% of ground truth box.

C. Experiment

The experiment's goal is to build Faster R-CNN model with Nie's proposed configuration and one improved Faster R-CNN model with the best configuration. Transfer learning is done by loading weights from pretrained ImageNet model before training the model with our dataset. Here is the experiment scenario.

1) Build the default config model

2) Build the modification config model

3) Every model is trained until 20 epoch for the RPN and until maximum 30 epoch for whole CNN. The final weight output is from an epoch with the least validation loss

4) Choose one model that has the best mAP value among all modification config model

Grid search technique is used to find the best configuration from experiment. The variation of hyperparameter value are shown by Table 1. The default value is the hyperparameter value proposed by Nie [4].

TABLE I. HYPERPARAMETER VARIATION FOR EXPERIMENT

Hyperparameter	Default Value	Custom Value
RPN *stride*	8	16
Optimizer	Adam	Adam, SGD
Learning rate	0.00001	0.001, 0.00001
Backbone CNN	VGG16	ResNet50

The training loss and RPN training accuracy chart of default configuration is shown by Fig. 5. After passing train phase, mAP is measured by evaluating model with data test from EMTD. Same thing happened to the other four custom models. It took 36 hours to train default configuration model. Plenty of hours are spent too just to train 4 custom models. Unfortunately, the free version of Google Colab only provides us 12 hours of machine runtime. So the runtime was paused and resumed until last epoch. From all custom models, The best modified model is picked by choosing a model with the best mAP value. Its training loss and RPN training accuracy chart is shown by Fig.6. Table 2 gives the brief of our experiment result.

TABLE II. EXPERIMENT RESULT

Backbone CNN	Optimizer	*Learning rate*	mAP	RPN training accuracy
VGG16	Adam	0.00001	**0.534**	0.964
ResNet50	Adam	0.00001	0.489	**0.979**
	Adam	0.001	N/A	0.961
	SGD	0.001	N/A	0.963
	SGD	0.00001	N/A	0.942

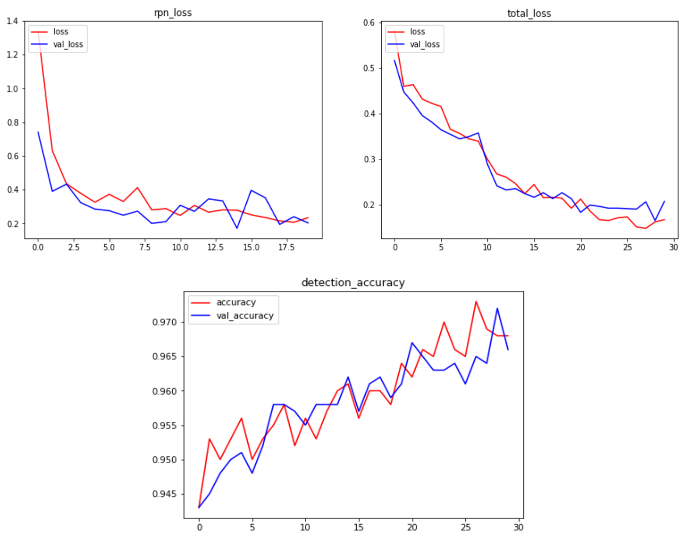

Fig. 5. Traning result of default configuration model

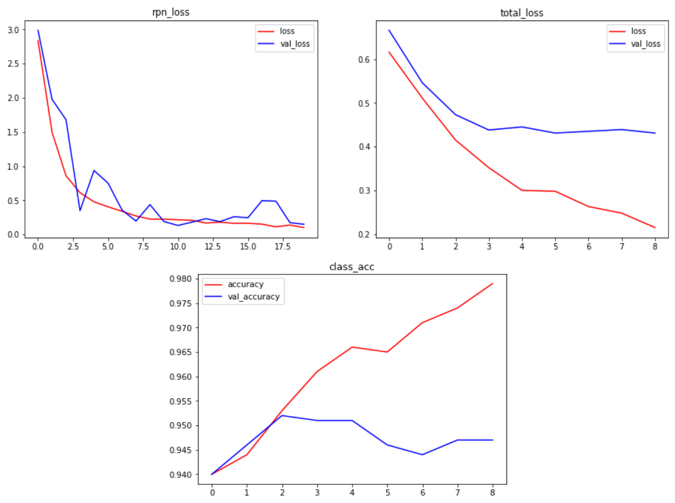

Fig. 6. Training result of modified model with the best configuration

The best mAP is given from default configuration model which gives 0.534 mAP. While the best RPN training accuracy is given from one of our custom models which gives 0.979 accuracy. The best modified configuration has ResNet50 as the backbone CNN, Adam as the optimizer, and 0.00001 learning rate. The other 3 models are failed to generate mAP because no true positive result could be generated. Potential size of object that can be detected is also observed. The optimum detected object size is 13x13 to 1156x1136 pixels for default configuration and 26x26 to 1156x1136 pixels for our modification model. Next, these two models will be tested on the road trip video.

D. Evaluation

This evaluation phase is aimed to compare the performance of the two models from experiment phase especially the frame processing speed. The data for evaluation is a road trip video which was taken by a dashcam at Jakarta toll road. Fig.7 depicts the footage of the evaluation video. The two models have been tested on the video. Default configuration model runs at 0.34 fps and our modified configuration model runs at 1.51 fps.

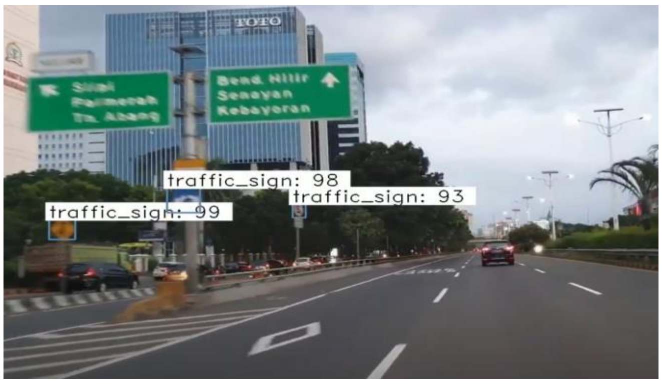

Fig. 7. Evaluation video's footage

Unfortunately, the two models still do not work perfectly on the video. Several false positives and false negatives can be found either. The examples are shown by Fig.8. False negatives are believed to be caused by the damaged traffic sign condition plus unclear representation of traffic sign that may be caused by the environment or the quality of the video. While false positives happen because the shape of the detected object resembles traffic sign such as signboard.

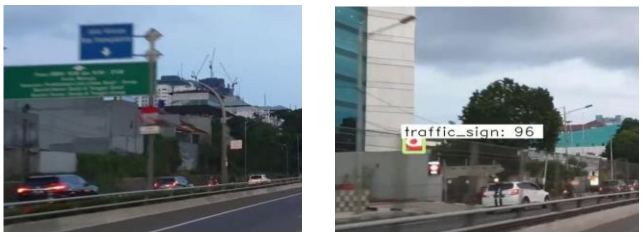

Fig. 8. Example of false negative (left) and false positive (right)

IV. Analysis

The two models we created work well in detecting Indonesian traffic sign. This is indicated by the result of our experiment which gives fair scores of mAP and by the result of our evaluation which gives very good detections. Though the accuracy of the two models has met our expectations, we found out that by changing the original configuration made by Nie, our modified model has its processing speed increased by 500%. The difference between the two models is only its base CNN which extracts the image's feature. VGG16 is replaced by ResNet50. As claimed by He [7], ResNet reduces the computational load done by VGGNet. Hence our modified model's speed outperforms the original one.

The model we created is supposed to run on autonomous vehicle system. That means it should fulfill the requirements of having high accuracy as well as high speed and is able to act in real time. While the result of the experiment shows that many improvements are still needed by our model to meet that ideal condition. Another method aside from Faster R-CNN should be tried in the future as it is not suitable for near real time detection. Furthermore, the quality of the video may affect the model performance. We only hard coded the frame extractor by utilizing OpenCV library. Our frame extractor may drop the video quality.

V. Conclusion and Future Work

Based on the experiment and evaluation results, it is concluded that traffic sign detection model built with Nie's configuration has better mean average precision value than our model that is built with modified configuration. However, our model is much faster by 5 times. Computational load of the feature extractor really impacts on the model processing speed. Although the speed of the model has been increased greatly, for an autonomous vehicle system our model is still not fast enough to be implemented with its real-time sensors. But generally, the two models work well in detecting Indonesian traffic signs.

For future work, more Indonesian traffic sign samples should be gathered, or it can be said, a dataset of Indonesian traffic sign should be used in training to improve model ability so that it can detect and classify what sign it detects. Moreover, using relevant dataset the evaluation becomes more representative for the real condition. Consider using a machine with large computational resource so that the limitation of resource doesn't hinder the experiment. The detection speed should be improved but the accuracy should not be reduced, so it is recommended to use another advanced method. We suggest using SSD or YOLO. It's also better to try to find the optimum speed of the vehicle that allows model to detect the traffic sign.

References

[1] Litman, T.A. "Autonomous Vehicle Implementation Predictions: Implications for Transport Planning". 2019.

[2] Wali, Safat & Abdullah, Majid & Hannan, M. A. & Hussain, Aini & Samad, Salina & Ker, Pin Jern & Mansor, Muhamad. "Vision-Based

Traffic Sign Detection and Recognition Systems: Current Trends and Challenges". Sensors. 19. 2093. 10.3390/s19092093. 2019

[3] Zuo, Z., Yu, K., Zhou, Q., Wang, X., & Li, T. "Traffic Signs Detection Based on Faster R-CNN. 2017 IEEE 37th International Conference on Distributed Computing Systems Workshops" (ICDCSW). doi:10.1109/icdcsw.2017.34, 2017.

[4] Nie, Jiaxi & Cheng, Yuan & Lan, Rui. "Traffic Sign Detection Based on Faster R-CNN". Academic Research Paper. University of Illinois, Urbana-Champaign, IL: Department of Electrical and Computer Engineering, 2017.

[5] Department of Transport. Know Your Traffic Signs Official Edition. London, UK: TSO, 2007.

[6] Ren, Shaoqing & He, Kaiming & Girshick, Ross & Sun, Jian. "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks". 1-10, 2016

[7] He, Kaiming & Zhang, Xiangyu & Ren, Shaoqing & Sun, Jian. "Deep Residual Learning for Image Recognition". 2015.

[8] Simonyan, Karen & Zisserman, Andrew "Very Deep Convolutional Networks for Large-Scale Image Recognition". 2015.

[9] Girshick, Ross. Fast r-cnn. 10.1109/ICCV.2015.169, 2015..