APPLICATION DEVELOPMENT OF VEHICLE COUNTING AND CLASSIFICATION USING YOLOV2

Yoga Nugraha Pryandika¹, Renny Pradina K¹

¹ Information System Department, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia

Email: yogapryandika@gmail.com

Abstract. In this research, we going to introduce a method named You Only Look Once v2 (YOLOv2) for vehicle detection and vehicle classification. The vehicle is going to classify is divided by 4 there are car, motorcycle, truck, and bus. The result of classified data it can provide data for Intelligent Transportation System (ITS) to engineering the traffic road to reduce the number of traffic jam in the big city. With this method we can receive higher accuracy and performance than another method proposed before. By using data consist of 21198 objects for training, from all that data there is consist of 8936 cars, 462 buses, 1681 trucks, and 10119 motorcycles. Then from all that data we can created some weight that took for 10 hours for training so that we can receive mean Average Precision (mAP) for 60.63 % mAP and higher performance detection with frame rate 20 up to 35 frame per seconds which is almost equal to real-time.

Keywords--Intelligent Transportation Systems, vehicle classification, vehicle detection, YOLOv2

1. Introduction

In this time, Growth of transportation sector have significantly increased especially in Indonesia. Based on the data from Central Bureau of Statistics in Indonesia the growth of vehicle since 2015 have significantly increased [1]. Every year the growth of vehicle in Indonesia increased 11.5 % in every year [2].

From all of that, many experts provide some solution to that problem, For example in Surabaya. They have a system called Surabaya Intelligent Transportation System (SITS). That system can provide traffic engineering to reduce traffic jams due to higher density in Surabaya [3]. In developing the system, surely need a detection and counting system to provide data for Intelligent Transportation System (ITS).

There are many researchers that proposed the different methods to detect and classify vehicle. Like as coarse-to-fine and boosted classifier. The most popular work used this method is viola and jones [4]. Another method to detect an object is Dictionary Based, example the best work of using this method is Bag of Word [5]. The last method is deep learning. One of the success works using deep learning is Convolutional Neural Network (CNN)[6]. For CNN there are a lot of type of CNN for object detection like R-CNN [7], Fast R-CNN [8] and Faster R-CNN [9]. From the method that mentioned before were proposed succesfully but it have a large network and detection speed is very low. So then in 2015 YOLO was proposed it have more detection speed and high accuracy[10].
In this research, we are going to use darkflow as a framework. For the method that will be used for detection and classification, we say You Only Look Once (YOLO), and we used the second version of YOLO or we can say YOLOv2 which can produce a higher level of precision and performance. YOLOv2 will be able to produce classification results that are more higher in performance and more higher accuracy than existing methods. And can produce a streaming video about the results of classification with low latency to less than 25 milliseconds [10].

2. Methodology

2.1. Datasets

The datasets we are going to use, is done by recording a video on the roads in the Surabaya, located on the waru highway, precisely on the Bungurasih pedestrian crossing bridge. The recording is divided into 6 positions namely front right, back right, front center, back center, left front, and left rear. The duration of each position is 10 minutes. Expected from 10 minutes to get 5000 images of each type. After the data is obtained, data processing will be carried out in the next process.

For pre-processing data is done by annotating the object in an image using the LabelImg Tool. LabelImg is a tool used to annotate objects in an image [11]. Then it can generate xml annotation files with PASCAL VOC 2007 format that can be used to conduct training on YOLOv2 architecture.

But before processing the video, the video that was recorded was converted into several images using FFmpeg. FFmpeg is a tool used to make a video into a collection of images [12]. Each picture will be taken every 2 seconds which is expected to have a different picture and from 1 hour the video will be divided into 40 minutes needed to conduct training data and 20 minutes to be used for testing.

2.2. Customize model configuration

After processing the dataset, we are going to use a library named darkflow to train our dataset. In darkflow there are consist our architecture we are going to use named YOLOv2. For the architecture of YOLOv2 represent in Table 1.

| Type          | Filters | Size/Side | Output |
|---------------|---------|-----------|--------|
| Convolutional | 32      | 3 × 3     | 224 × 224 |
| Maxpool       | 2 × 2   | 112 × 112 |        |
| Convolutional | 64      | 3 × 3     | 112 × 112 |
| Maxpool       | 2 × 2   | 56 × 56   |        |
| Convolutional | 128     | 3 × 3     | 56 × 56  |
| Convolutional | 64      | 1 × 1     | 56 × 56  |
| Convolutional | 128     | 3 × 3     | 56 × 56  |
| Convolutional | 256     | 3 × 3     | 56 × 56  |
| Maxpool       | 2 × 2   | 28 × 28   |        |
| Convolutional | 512     | 3 × 3     | 28 × 28  |
| Convolutional | 256     | 1 × 1     | 28 × 28  |
| Convolutional | 512     | 3 × 3     | 28 × 28  |
| Maxpool       | 2 × 2   | 14 × 14   |        |
| Convolutional | 512     | 3 × 3     | 14 × 14  |
| Convolutional | 256     | 1 × 1     | 14 × 14  |
| Convolutional | 512     | 3 × 3     | 14 × 14  |
| Maxpool       | 2 × 2   | 7 × 7     |        |
| Convolutional | 512     | 3 × 3     | 7 × 7    |
| Convolutional | 1024    | 1 × 1     | 7 × 7    |
| Convolutional | 512     | 1 × 1     | 7 × 7    |
| Convolutional | 1024    | 3 × 3     | 7 × 7    |
| Convolutional | 1024    | 3 × 3     | 7 × 7    |

Table 1. Architecture of Darknet 19

In Table 1, the architecture of YOLOv2 used CNN as the basis. Then the CNN in YOLOv2 is modified in the flatten part. Flatten in YOLOv2 is removed and the detection and classification of an object not using fully connected anymore but replaced with K – Means classification for improve accuracy of the model.
In YOLOv2 there are consist of 19 convolutional layer and 5 \textit{maxpool} layers[13]. In the next step we do some customization in the last convolutional layer, we change the number of filter we going to use with function (1)[14].

\[ m_i \ast (c + \hat{c}) \]  

(1)

For \( cls \) in function (1) is several classes we going to use. In this case the number of \( cls \) we going to use is 4. Next is \( coord \). \( Coord \) is represent of several things contained in each bounding box there are \((x,y)\), \((w,h)\) and \(C\) so the number of \( coord \) is 5. For the \( num \) is the number of bounding boxes in each grid which is 5 so the total number we going to use is \( 5 \ast (4 + 5) = 45 \).

2.3. \textit{Training function}

During we do training our model. Loss function in the model divided by three loss that is classification loss, localization loss, confidence loss[10]. For the loss function can be seen in function (2)

\[
\begin{align*}
\lambda_c \sum_{i=0}^{w-1} \sum_{j=0}^{h-1} & \left( (x_i - x_i^p)^2 + (y_i - y_i^p)^2 \right) + \lambda_b \sum_{i=0}^{w-1} \sum_{j=0}^{h-1} \left( \sqrt{w_i} - \sqrt{w_i^p} \right)^2 + \left( \sqrt{h_i} - \sqrt{h_i^p} \right)^2 \\
& + \sum_{i=0}^{w-1} \sum_{j=0}^{h-1} \left( t_i - t_i^p \right)^2 + \lambda_j \sum_{i=0}^{w-1} \sum_{j=0}^{h-1} \left( t_i - t_i^p \right)^2 \\
& + \sum_{i=0}^{w-1} \sum_{j=0}^{h-1} (\hat{c} - \hat{c}_i)^2
\end{align*}
\]  

(2)

In function (2) for \( p \) is for classification loss and for \( 1^0 \), if there is an object the that cell has an value 1 and if not it has a value zero. The next is \textit{localization loss}, in the localization loss it measures error when predicting the location of bounding box and its size it represented by \((x,y)\) is about calculate the width and height of predicted bounding box. Usage of square root in this function due to this function must be reflect small deviations in large boxes matter less than in small boxes. So, to partially address this by adding a square root of the width and height of bounding box instead of the width and height directly. To compute the loss about their confidence for each bounding box prediction. \( C \) is for confidence score and \( \hat{c} \) is the intersection over union of the predicted bounding box and the ground truth. \( \text{1obj} \) is represent for if in that cell there is have an object then is equal to one, and 0 otherwise. And for \( \text{1noobj} \) is the opposite.

2.4. \textit{Counting systems}

In this research for counting system using the result of detection. The result of detection is divided by 30, Because the frame rate of video is 30 FPS. Then the result of that calculation putted in some list in python and displayed with \textit{matplotlib}.

3. \textit{Results}

3.1. \textit{Training}

Dataset is classified based on government regulation no.55 year 2012[15] tells about the type of the vehicle. Vehicles are classified into 4 types namely motorbikes, passenger cars that have a maximum seat of 8, Bus Cars are motorized vehicles transporting people who have more than 8 people, and the last is a goods car vehicle, which is designed in part or in whole to transport goods. And the four types of vehicles if simplified into cars, buses, and trucks.

From all four classifications from overall images there are 21198 objects. With among them 8936 cars, 462 buses, 1681 trucks and 10119 motorcycles. Dataset recorded with resolution 1440 x 1080 pixels with frame rate 25 FPS and this is an example from images represent in Figure 1.
This research has been performed in platform with GPU Nvidia GeForce RTX 2060 6GB, using Ubuntu 16.04 with Core i7 4790 CPU and a RAM of 8 GB and for the configuration we going to use the percentage of GPU used is 80% and for epoch is 500 and for batch is 10.

With that configuration, the training took up to 10 hours. In every process resulting loss in every iteration. Loss will eventually be decreased in every iteration. Model trained with learning rate 0.001 and momentum 0.9 and Leaky Rectified Linear Unit (ReLU) is used as activation function. Moreover, popular algorithm like Stochastic Gradient Descent (SGD) is applied for updating learning parameter until convergent. With all that configuration result of this experiment is represent in Figure 2.

Along with decreased loss. Accuracy will eventually be increased in every iteration. For the result of accuracy can get 84.90% and for loss is 15.11%

3.2. Validation
The number of data is used to do some testing in this experiment can be seen in Table 4.

| Type    | Total |
|---------|-------|
| Car     | 2089  |
| Bus     | 123   |
| Truck   | 350   |
| Motorcycle | 1987 |

The result of training is in the form of weights. Then the weight is tested again with data represent in Table 4. The unit we going to use to represent our validation result is Mean Average Precision (mAP) and performance is Frame Per Second. Mean Average Precision is defined by calculation from Precision (P) and Recall (R).
True positive and false positive is obtained from IoU from tested data. True Positive if a data received IoU more than 0.5, false positive is the opposite. From result of true positive and false positive mAP can be obtained. For the mAP and Average Precission for each class can be seen in Figure 5.

The configuration of threshold is used in this experiment is 0.5. with that configuration, resulting 60.63 % mAP, bus get the highest value of AP with 0.87 AP and for the lowest AP is motorcycle with 0.46 AP. And then testing is done at various thresholds to see, on how much threshold this detector can get best mAP for the result, can be seen in Table 5.

| threshold | Motor | Car | Bus | truck | mAP |
|-----------|-------|-----|-----|-------|-----|
| 0.3       | 0.48  | 0.60| 0.89| 0.62  | 64.85%|
| 0.4       | 0.47  | 0.58| 0.88| 0.58  | 62.95%|
| 0.5       | 0.46  | 0.57| 0.87| 0.53  | 60.63%|
| 0.6       | 0.44  | 0.55| 0.84| 0.47  | 57.59%|
| 0.7       | 0.39  | 0.51| 0.80| 0.40  | 52.50%|

From Table 3 we can see that if we use higher threshold will be resulting more lower mAP. But if we adjust the threshold more lower the result of mAP will be more higher. but if we use lower threshold it will impact to false positive and true positive. In shown how development of true positive and false positive in every threshold.

![False Positive and True Positive of Motorcycle in every threshold](image)

In Figure 5 can be seen that, the lower threshold we used in experiment the result is higher false positive and so with true positive.

For performance of YOLOv2 the frame rate we can get from YOLOv2 is 35 fps. With that number it demonstrate the real time performance. In Figure 6 show the result how detector classify and detect vehicle.
4. Conclusions

YOLOv2 have an outstanding accuracy and performance to detect and classifying vehicles. Accuracy and performance YOLOv2 can get is 60.63 % mAP in threshold 0.5 and frame rate up to 35 fps. With this frame rate we can achieve almost real time detection. But YOLOv2 have a problem with lower thresholds. If we use lower threshold, mAP will be getting higher but false positive will be higher too. To overcome this problem the threshold we going to use must be adjusted with what we going to use. If we want more higher accuracy use higher threshold, but if want the detector to detect amount of object, we must balance the threshold of our detector between false positive and true positive.

References

[1] Badan Pusat Statistik, “Perkembangan Jumlah Kendaraan Bermotor Menurut Jenis, 1949-2016.” [Online]. Available: https://www.bps.go.id/linkTableDinamis/view/id/1133. [Accessed: 14-Jan-2019].

[2] R. Pratama, “Jumlah Kendaraan Kian Banyak, Membanggakan Tapi bikin Nggak Sehat,” 2018. [Online]. Available: https://oto.detik.com/mobil/d-4242211/jumlah-kendaraan-kian-banyak-membanggakan-tapi-bikin-nggak-sehat. [Accessed: 16-Jan-2019].

[3] P. Maulidya, “Kemacetan di Surabaya Bisa Dipantau Lewat Aplikasi SITS - Surya,” 2018. [Online]. Available: http://surabaya.tribunnews.com/2018/10/11/kemacetan-di-surabaya-bisa-dipantau-lewat-aplikasi-sits. [Accessed: 15-Jan-2019].

[4] P. Viola and M. Jones, “Robust Real-Time Face Detection,” Int. J. Comput. Vis., vol. 57, no. 2, pp. 137–154, 2004.

[5] J. Mutch and D. G. Lowe, “Object class recognition and localization using sparse features with limited receptive fields,” Int. J. Comput. Vis., vol. 80, no. 1, pp. 45–57, 2008.

[6] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” Proc. IEEE, vol. 86, no. 11, pp. 2278–2324, 1998.

[7] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation,” in 2014 IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 580–587.

[8] R. Girshick, “Fast R-CNN,” in 2015 IEEE International Conference on Computer Vision (ICCV), 2015, vol. 2015 Inter, pp. 1440–1448.

[9] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 39, no. 6, pp. 1137–1149, 2017.

[10] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You Only Look Once: Unified, Real-Time Object Detection,” 2015.

[11] “LabelImg,” 2017. [Online]. Available: https://devhub.io/repos/tzutalin-labelImg. [Accessed: 14-Feb-2019].
[12] FFmpeg. “About FFmpeg.” [Online]. Available: https://www.ffmpeg.org/about.html. [Accessed: 15-Feb-2019].
[13] J. Redmon and A. Farhadi, “YOLO9000: Better, Faster, Stronger,” Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017, vol. 2017-Janua, pp. 6517–6525, Dec. 2016.
[14] M. Wang, M. Liu, F. Zhang, G. Lei, J. Guo, and L. Wang, “Fast Classification and Detection of Fish Images with YOLOv2,” in 2018 OCEANS - MTS/IEEE Kobe Techno-Oceans (OTO), 2018, pp. 1–4.
[15] “PERATURAN PEMERINTAH REPUBLIK INDONESIA NOMOR 55 TAHUN 2012 TENTANG KENDARAAN,” 2012.