Vehicle Detection with Sub-Class Training using R-CNN for the UA-DETRAC Benchmark

Sitapa Rujikietgumjorn Nattachai Watcharapinchai
National Electronics and Computer Technology Center (NECTEC), Thailand
{sitapa.rujikietgumjorn,nattachai.watcharapinchai}@nectec.or.th

Abstract

Different types of vehicles, such as buses and cars, can be quite different in shapes and details. This makes it more difficult to try to learn a single feature vector that can detect all types of vehicles using a single object class. We proposed an approach to perform vehicle detection with Sub-Classes categories learning using R-CNN in order to improve the performance of vehicle detection. Instead of using a single object class, which is a vehicle in this experiment, to train on the R-CNN, we used multiple sub-classes of vehicles so that the network can better learn the features of each individual type. In the experiment, we also evaluated the result of using a transfer learning approach to use a pre-trained weights on a new dataset.

1. Introduction

Vehicle detection is one of the essential process for many vision-based traffic surveillance applications. The location or a bounding box of a vehicle must be extracted from the traffic image. This detected location of a vehicle in an image can be further applied to several applications such as vehicle tracking or counting. The cropped image of a vehicle can also be used for vehicle type and model classification.

Several challenges occur in vehicle detection domain. Occlusion is one major challenge that can largely reduce the detection performance especially when a vehicle is highly occluded. Usually, the videos from traffic surveillance systems are acquired from the practical environment and the weather and lighting condition can also affect the detection performance. Shadows from vehicles and other objects can also be as obstacle to the detection as well. The variations in shapes from different orientations also make vehicle detection more difficult. For example, the front side is visually different than the side of the vehicle.

Vehicle is a rigid object and several features have been used in vehicle detection such as Aggregate Channel Features (ACF) [3]. Since vehicle has many unique parts, part-based features such as Deformable Part Model (DPM) [4], or Linked Visual Words [9] are also used for vehicle detection and classification. Recently, convolutional neural network is widely used for object classification and detection and it has been applied to vehicle detection and classification [11].

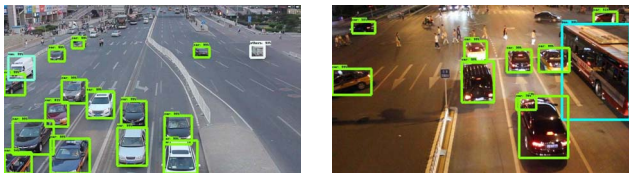

Figure 1: Examples of detected results

In this paper, the convolutional neural network is used for vehicle detection based on R-CNN architecture. We propose to use multiple sub-classes of vehicles instead of using a single vehicle class for training. A transfer learning approach is also evaluated for fine tuning a pre-trained model to be used with a traffic surveillance dataset.

2. Transfer Learning for the Convolutional Neural Network

Training the convolutional neural network from scratch takes time and it needs a sufficiently large dataset. An insufficient amount of dataset may lead to overfitting problem. On the other hand, several pre-trained models are available but it may not be feasible to directly apply to another dataset. Figure 2 shows detection results on UA-DETRAC dataset when directly using the model trained on COCO dataset [7] that has 90 object classes including vehicles. There are still a lot of missed detections in the result and some incorrect detections such as some cars were classified as boats or cell phones. Nevertheless, these pre-trained weights are still useful and can be used for weight initialization and fine tuning the network on a new dataset using a transfer learning approach.

978-1-5386-2939-0/17/$31.00

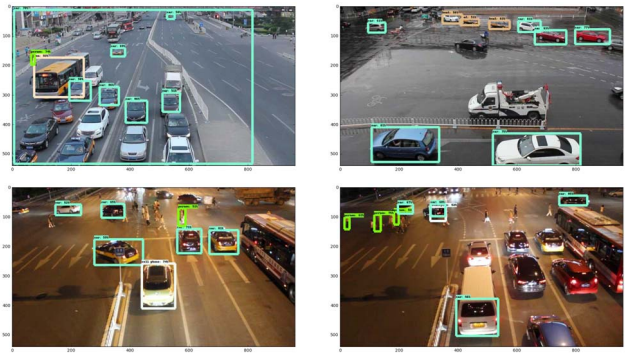

Figure 2: Examples of vehicle detection results using models trained on the COCO dataset without transfer learning

Two main factors that will affect the selection of transfer learning approach are the dataset size and the similarity of the new dataset to the dataset used in the pre-trained network. There are four major scenarios for transfer learning and fine tuning.

Small and similar dataset
When the dataset is small, it is not good to fine tune the weights as it may result in overfitting problem. The CNN can be used as a feature extractor since the higher level features of the network are still relevant to the new dataset [12]. And a linear classifier is trained on this CNN features instead.

Small and different dataset
Since the new dataset is different, the higher level features, which are more specific to the original dataset, may not be suitable. It is better to use the lower level features from the early layers because the features are more generic and train the linear classifier on these lower level features.

Large and similar dataset
With sufficient data, we can fine-tune the weights of the pre-trained network via backpropagation. The pre-trained weights are used as weight initialization for fine tuning to the new data set.

Large and different dataset
Since the dataset are different and there are sufficient data, we can train the network from scratch. However, it may still be beneficial to initialize the weights from the pre-trained model instead of using random weight initialization.

The UA-DETRAC dataset size is large enough for fine tuning and some classes in the COCO dataset are similar to the UA-DETRAC dataset. The COCO dataset has 90 object categories including vehicle classes which are car, bus, truck, motorcycle, and bicycle. In contrast, the UA-DETRAC dataset has four vehicle classes in the annotation which are car, bus, van, and other. For our approach, we use the pre-trained weights as an initialization to fine tune the network to a new dataset. These pre-trained weights were trained using the COCO dataset. Since our test dataset (UA-DETRAC) is similar to some categories in the COCO dataset, we used a COCO pre-trained model [6] for fine tuning the full network with UA-DETRAC dataset.

3. Sub-Classes Training Using R-CNN

Our algorithm is based on the Faster R-CNN architecture [8] with residual network. In our experiment, we fine-tuned R-CNN with 101-layers residual network on the UA-DETRAC dataset for vehicle detection. Instead of using one output class of vehicle, we sub-categorized the vehicle class into four classes which are car, van, bus, and others. The example images of these four sub-classes are shown in Figure 3. The sub-class is a prior knowledge to help the single object detection in which the object is quite differentiated in shapes and details. This sub-classes technique should help the network to better learn the features of each different vehicle types.

(a) Car (b) Van (c) Bus (d) Other

Figure 3: Example images of four sub-classes of vehicles

In addition to vehicle types, orientation of the vehicles can also be quite different such as frontal and side view. These sub-classes can also be used as a prior knowledge as well. Figure 4 shows example images of eight vehicle orientations we used in the experiment.

Figure 4: Example images of eight vehicle orientations

4. Experimental Results

In our experiment, the algorithms were evaluated according to the submission rules as specified in the UA-DETRAC benchmark [10]. In this section, we give a detailed analysis of comparison between our proposed method and the baseline methods.

4.1. Dataset

The dataset we used in the experiment is UA-DETRAC dataset, a real-world multi-object detection and multi-object tracking benchmark. This dataset is a 10-hours traffic video sequence at various challenging levels such as variations in scale, pose, illumination, night time and day time, occlusion, and background clutters. The dataset is divided into a training and test set with 60 and 40 sequences respectively.

In our experiment, we divided the original training set into a training and validation set. Four sequences from the training set were selected as our validation set. The selected sequences were `MVI_39851`, `MVI_39861`, `MVI_40161`, and `MVI_40131`. In the 2017 UA-DETRAC challenge, the test set was divided into two levels, beginner and experienced. We participated in the beginner level in which there were 10 test video sequences. The test set must be evaluated through the UA-DETRAC challenge submission server.

4.2. Experimental Environment

The hardware specification in the experiment was 2x Intel Xeon E5-2630v3 with RAM 384GB, and GPU 2x Nvidia Tesla K80. The operating system was Ubuntu 16.04 (64bits). The code was developed using python 2.7 and Tensorflow [1].

4.3. Evaluation Metric

Average precision (AP) is used for our detection evaluation. The proposed method with various variations were evaluated on the validation set comparing to the baseline methods. This validation set was used for our evaluation among different variation of our proposed methods to select the best method for submitting to the UA-DETRAC challenge submission server for further evaluation on the test set.

4.4. Detection Results

Our proposed method is compared with four baseline methods which are DPM [4], ACF [3], R-CNN [5], and CompACT [2]. The average precision score for the validation set is shown in Table 1. For our method, we evaluated our R-CNN that used transfer learning (TL) with one class and several sub-classes. For the 4 sub-classes, the types of vehicles are assigned to outputs of R-CNN such as car, van, bus, and other. To further evaluate the performance when increasing the number of sub-class, we also use 4 sub-classes and 8 vehicle orientations, resulting in 32 sub-classes in total. The result shows that our R-CNN with 4 sub-classes improves the R-CNN with one vehicle class by almost 5%. However, the performance decreases when using 4 sub-classes and 8 vehicle orientations. Using too many sub-classes with similar features, like a slightly different view of vehicle in this case, does not improve the performance. So, the sub-class should be different enough to gain the performance.

Method	Validation Set
DPM [4]	29.21%
ACF [3]	72.01%
R-CNN [5]	74.36%
CompACT [2]	76.43%
R-CNN + TL with 1 class	88.89%
R-CNN + TL with 4 sub-classes	**93.70%**
R-CNN + TL with 4 sub-classes and 8 orientations	90.08%

Table 1: Average precision (AP) scores of all comparison methods on the validation set.

In the experiment, we also compared our method that used transfer learning of the model trained on an especially large dataset with lots of object classes and one that was trained from scratch (no transfer learning). The result in Table 2 shows that average precision is much higher when using a transfer learning from a good model to fine tune to a new dataset compared to a new training on a new dataset.

One issue of using multiple sub-classes instead of single class for single object detection is that the detection bounding boxes of different sub-classes are overlapped as shown in Figure 5 left. From Table 1, the performance of using 4 sub-classes and 8 orientations decreases compared to using just 4 sub-classes which is the result of increasing in false positives. However, in the experiment, performing non-maximum suppression (NMS) to remove the overlapped bounding box does not increase the detection performance. Table 3 shows slight decreasing in the average precision when performing non-maximum suppression on the case of 4 sub-classes and 8 orientations. The reason of performance decreasing is because the non-maximum suppression also removes the correct bounding boxes being overlapped as shown in the yellow bounding box in Figure 5 right.

The evaluated performance of our R-CNN with four sub-classes on the test set is shown in Table 4. This result was evaluated by UA-DETRAC submission server. For the beginner level, the test set does not include the medium and hard sequences. Our proposed method outperforms the baseline methods and our overall performance is 93.43%. Various conditions are also separately evaluated such as cloudy, night, rainy, and sunny. Figure 6 shows detection results on the test set. Different bounding box color indicates a different sub-class. For row 3, all results in the transfer learning approach (column2-4) have a traffic sign incorrectly detected as a vehicle. However, the scratch training (column 1) does not have this incorrect detection. The result image in row 4 column 3 shows that the multiple types

Method	Validation Set
R-CNN + no TL with 1 class	57.08%
R-CNN + TL with 1 class	88.89%
R-CNN + no TL with 4 sub-classes and 8 orientations	36.31%
R-CNN + TL with 4 sub-classes and 8 orientations	90.08%

Table 2: Average precision (AP) scores of results comparing our methods that use transfer learning (TL) and ones without transfer learning

Method	Validation Set
R-CNN + TL with 4 sub-classes and 8 orientations	90.08%
R-CNN + TL with 4 sub-classes and 8 orientations + NMS	88.23%

Table 3: Average precision (AP) scores of results comparing our methods with and without non-maximum suppression algorithm

Figure 5: Example cases of multiple overlapped detection results on the same object (left) and multiple overlapped detection results on different objects (right).

of detecting vehicles are assigned on the same object.

5. Conclusions

Sub-Classes categories learning using R-CNN to improve the performance of vehicle detection is presented in this paper. Instead of using a single vehicle class for vehicle detection, we use multiple sub-classes of vehicles so that the R-CNN can better learn the features of each individual type. Nevertheless, the sub-class should be different enough to gain the performance since using too many sub-classes with similar features, such as vehicle orientation, does not improve the performance. There are several approaches for transfer learning the weights to a new dataset. In the experiment, we compared the result of using transfer learning with the result of training from scratch. The result shows that the transfer learning has a better performance.

There is still an issue of multiple bounding box overlap from different sub-class on the same object. As a future work, the network should be modified and improved such that a single sub-class type is outputted in a particular region. The full overlap of bounding box but only partial occlusion on the object is another interesting point that may be improved by using a better bounding box suppression method.

References

[1] M. Abadi and et. al. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] Z. Cai, M. Saberian, and N. Vasconcelos. Learning complexity-aware cascades for deep pedestrian detection. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 3361–3369, Dec 2015.

[3] P. Dollr, R. Appel, S. Belongie, and P. Perona. Fast feature pyramids for object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(8):1532–1545, Aug 2014.

[4] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, Sept 2010.

[5] 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*, pages 580–587, June 2014.

[6] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, and K. Murphy. Speed/accuracy trade-offs for modern convolutional object detectors. *CoRR*, abs/1611.10012, 2016.

[7] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014.

[8] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Neural Information Processing Systems (NIPS)*, 2015.

[9] N. Watcharapinchai, S. Aramvith, and S. Siddhichai. Automatic vehicle classification using linked visual words. *Journal of Electronic Imaging*, 26(4):043009, 2017.

[10] L. Wen, D. Du, Z. Cai, Z. Lei, M. Chang, H. Qi, J. Lim, M. Yang, and S. Lyu. UA-DETRAC: A new benchmark and protocol for multi-object tracking. *CoRR*, abs/1511.04136, 2015.

[11] L. Yang, P. Luo, C. C. Loy, and X. Tang. A large-scale car dataset for fine-grained categorization and verification. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3973–3981, June 2015.

[12] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? *CoRR*, abs/1411.1792, 2014.

Method	Test Set							
	Overall	Easy	Medium	Hard	Cloudy	Night	Rainy	Sunny
DPM [4]	25.70%	34.42%	30.29%	17.62%	24.78%	30.91%	25.55%	31.77%
ACF [3]	46.35%	54.27%	51.52%	38.07%	58.30%	35.29%	37.09%	66.58%
R-CNN [5]	48.95%	59.31%	54.06%	39.47%	59.73%	39.32%	39.06%	67.52%
CompACT [2]	53.23%	64.84%	**58.70%**	**43.16%**	63.23%	46.37%	44.21%	71.16%
R-CNN + TL with 4 SC (ours)	**93.43%**	**93.43%**	N/A*	N/A*	**96.69%**	**92.54%**	**87.30%**	**94.47%**

Table 4: Average precision (AP) score of the results on the test set from the UA-DETRAC submission server (*The beginner level does not include the medium and hard test set).

(a) R-CNN + no TL with 1 class

(b) R-CNN + TL with 1 class

(c) R-CNN + TL with 4 sub-classes

(d) R-CNN + TL with 4 sub-classes and 8 orientations

Figure 6: Detection result images on the test set. Different bounding box color indicates a different sub-class.