Vehicle Classification in Video Using Deep Learning

Mohammad O. Faruque, Hadi Ghahremannezhad, and Chengjun Liu

New Jersey Institute of Technology
Department of Computer Science
Newark, NJ 07102 USA

Abstract. Vehicle classification in videos has broad applications in intelligent transportation and smart cities. The vehicle classes are defined according to the Federal Highway Association (FHWA) vehicle types, and two popular deep learning methods, namely, the Faster R-CNN and the YOLO, are applied for vehicle classification. The Faster R-CNN and the YOLO are two representative deep learning methods with applications in object detection and classification. First, three training data sets are manually created from two videos in the low video quality category for training the Faster R-CNN and the YOLO deep learning methods. Second, new videos that are not seen during training are used to evaluate the vehicle classification performance for the deep learning methods. In particular, the comparative evaluation includes the training time, the testing time, the vehicle classification accuracy, as well as the generalization performance of the deep learning methods. The experiments using the New Jersey Department of Transportation (NJDOT) traffic videos show the feasibility of vehicle classification in videos using deep learning methods.

Keywords: Vehicle classification, intelligent transportation, smart cities, deep learning, Faster R-CNN, YOLO.

1 Introduction

Classifying vehicles in traffic video into different categories has broad applications in intelligent transportation and smart cities. Based on the Federal Highway Association (FHWA) vehicle types, we define six vehicle categories: bike, car, truck, van, bus, and trailer (Fig.1). Vehicle classification in the video, therefore, classifies the vehicles into these predefined six vehicle categories.

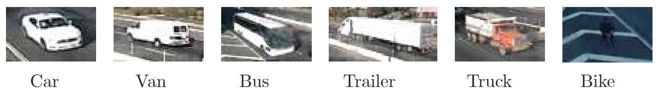
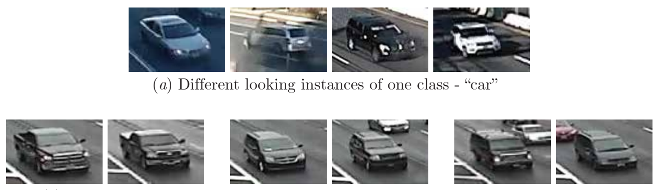


Fig. 1. Six types of vehicles used for classification.

There are various challenging issues in vehicle classification from the video due to the camera viewing directions, the illumination variations, and weather conditions. Fig. 2 shows some challenges for vehicle detection. The images in Fig. 2 are all collected from the New Jersey Department of Transportation (NJDOT) traffic video sequences. Note that these videos are captured from similar camera angles during day time. In general, there can be more different visual scenarios, and the vehicle detection problem may become more challenging. Changes in environmental conditions, the background of objects, time of the day, occlusions, blur, motion, and camera resolution all make the vehicle detection task more challenging.

We have applied two representative deep learning methods, namely, the Faster R-CNN and the YOLOv3 deep learning methods [23], [22], for vehicle classification in the video. We have used eight traffic videos with the encoding quality of 15fps frame rate and 352x240 spatial resolution to evaluate the vehicle classification performance of deep learning methods. For the training data, we have collected training samples from two traffic videos (video 1 and video 4) in which we manually annotate the vehicles. These training samples are used to define three training data sets corresponding to the two traffic videos: video 1, video 4 and a mixed set of video 1 and video 4. These different data sets thus help us evaluate the comparative generalization performance of the two deep learning methods. For testing, the videos that are not seen during training are applied to evaluate the testing time, the vehicle classification accuracy, as well as the generalization performance of the deep learning methods.



(*a*) Different looking instances of one class - "car"

(*b*) Similar looking instances of different classes - "pickup truck", "car" and "van"

Fig. 2. Some visual challenges in object classification. (a) Various instances of one class with differences in angle, color, size, and other visual attributes. (b) Similar objects from different categories. A good detection method should be able to detect the trivial differences to categorize objects correctly.

2 Background

Object detection and classification has been a popular topic in computer vision and video analysis. Many methods have been published in the literature using statistical methods, such as Support Vector Machine (SVM) [19], efficient SVM (eSVM) [3], clustering-based discriminant analysis [2], the Bayesian Discriminating Features (BDF) method [13], and Adaboost [26]; local features methods, such as Local Binary Patterns (LBP) [18], Scale Invariant Feature Transform (SIFT) [16], Histogram of Oriented Gradient (HOG) [4], and Feature Local Binary Patterns (FLBP) [9]; neural networks and deep learning methods [24], [14]. More recently, object detection and classification methods based on deep learning have had a good amount of success in many competitions, such as the ILSVRC large scale detection challenge [5], the PASCAL VOC detection challenge [7][6], and the MS COCO large scale detection challenge [12].

The Convolutional Neural Network (CNN) is one of the popular Deep Neural Network (DNN) architectures. DNN usually denotes a feed-forward artificial neural network that has multiple hidden layers between the input and output layers [1]. Convolutional Neural Networks are a specific type of DNNs that are feasible for large input data with locally meaningful connected patterns like images. During each forward pass in a CNN, each convolutional layer extracts features utilizing the learned convolution filters and by updating the weights learned through the training process. The convolution layers are usually followed by activation layers like Rectified Linear Unit (ReLU) [17] (which gets rid of the negative values) and pooling layers.

The Region based CNN (R-CNN) uses region proposals through selective search [25] that applies different window sizes to evaluate the entire image. First, in the selective search process, it extracts around 2000 regions. Then, the R-CNN applies a custom version of the AlexNet [10] to determine a valid region. At the final fully connected layers, it utilizes a number of binary support vector machines to classify the objects.

The Fast R-CNN, which was proposed in 2015 [8], improves upon the R-CNN as R-CNN is slow due to the need of a forward pass for each proposed region individually, The Fast R-CNN is faster than R-CNN because of combining different parts of the process and sharing computations. Since the region proposals of each image have a high overlap with each other, this approach tries to share the convolution calculations along the network layers. In each forward pass, the entire image and all its region proposals are fed to the CNN at the same time. The region proposals of each image share the generated feature maps along the network layers, and thus the speed of the model improves by reducing the time of computations. For this to happen, the last max-pooling layer of the pre-trained CNN is replaced with a Region of Interest Pooling (RoI-pooling) layer which takes region proposals with different sizes and outputs fixed-length feature vectors.

The Faster R-CNN [23], which improves upon the Fast R-CNN, was introduced in 2016, and it combines the region proposal and the CNN modules [8]. The Faster R-CNN eliminates the ”selective search” algorithm and instead uses another network called Region Proposal Network (RPN) to generate object proposals and also learn from the Fast R-CNN network [8].

You Only Look Once (YOLO) deep learning method was introduced in 2015 [20]. In this approach, an input image is divided into an $S \times S$ grid, and each cell in the grid is used to detect only one object (in case there exists one) whose center falls in that cell. Fig. 4 shows the idea of vehicle detection using YOLO.

In each cell, a fixed number (B) of bounding boxes with their confidence scores is generated. The confidence scores are calculated by multiplying the probability of each object and their intersection over the union of the predicted box and the ground truth box.

YOLOv2 is an improved version upon YOLO, as the error analysis of YOLO showed that it does not perform well in several cases, such as producing a significant number of localization errors and having relatively low recall compared to region proposal-based methods [21]. YOLOv2 introduces several improvements upon YOLO, such as batch normalization, high-resolution classifier, dimension clusters, direct location prediction, and multi-scale training. Darknet-19 was used as the base classification model of YOLOv2. Like VGG models it mostly uses 3×3 filters, and after every pooling step, it doubles the number of channels. In total, it uses 19 convolutional layers and five max-pooling layers.

YOLOv3 is introduced as an incremental update [22]. To integrate well with the Open Images dataset, it replaces the softmax layers with the independent logistic classifiers and uses binary cross-entropy loss for the class predictions during training. Other changes include using a new classifier, using Darknet-53 instead of Darknet-19, and making detections in three different scales. These changes help YOLOv3 to identify small objects. Finally, it is claimed that YOLOv3 is three times faster than the SSD method with a similar accuracy to the SSD [15] and the RetinaNet [11] models.

3 Vehicle Classification in Video Using Deep Learning

We apply two representative deep learning methods, the Faster R-CNN, and the YOLOv3, for vehicle classification in the video. The training data sets are manually created using two NJDOT traffic videos. Specifically, three training data sets are created corresponding to the training samples from video 1, video 4, and a mixed set of video 1 and video 4, respectively.

3.1 Vehicle Classification in Video Using the Faster R-CNN Deep Learning Method

We apply the Faster R-CNN deep learning method for vehicle classification in the video. The VGG16 network is used with the Faster R-CNN deep learning method, and Fig. 3 shows the system architecture of the Faster R-CNN model. Note that the Region Proposal Network (RPN) network works proportionally to the Fast R-CNN network and uses it to generate better region proposals in the process of training [8]. In this version, the RPN is fine-tuned by pre-trained convolution network on image classification task. In this phase, positive samples are the ones with the Intersection over Union (IoU) more than 0.7, and negative samples are the ones with IoU less than 0.3, and the rest of the samples stays ignored. In this module, a small $n \times n$ (by default n=3) window is slid over the feature map of the entire image. At the center of each sliding window, nine anchors with three different scales and three different ratios are generated. Then

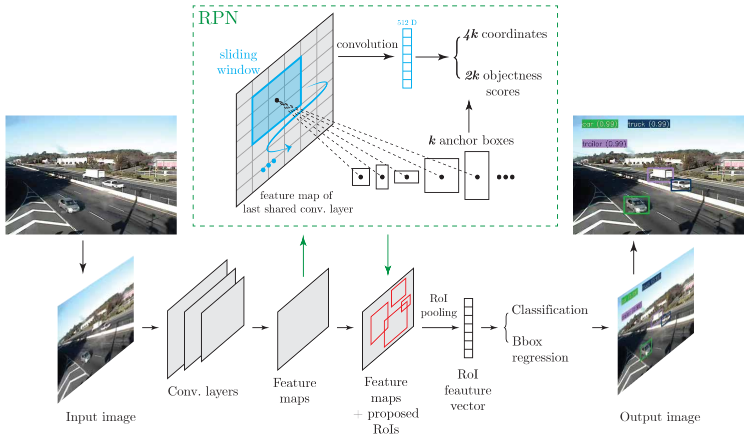


Fig. 3. The system architecture of the Faster R-CNN model. The RPN network works proportionally to the Fast R-CNN network and uses it to generate better region proposals in the process of training.

each anchor is fed to classifiers and bounding box regressors to be classified as foreground or background and also to be refined. These anchors are the region proposals that are used to train the Fast R-CNN model, and simultaneously, the output of the Fast R-CNN is used to initialize the training of the RPN. These two networks share convolutional layers, and therefore, this approach becomes faster than the Fast R-CNN with the selective search. After RPN generates the region proposals, these regions have different sizes, and they are fed to CNN after RoI-pooling.

In particular, the VGG16 network has a total of 13 convolutional layers, five max-pooling layers, three fully connected layers, and one softmax classification layer. The size of convolution filters applied to the feature maps is by default 3×3, and the size of max-pooling layers is 2×2. The rectification non-linear activation is applied to all the hidden layers of this network. In the pooling layers, the act of down-sampling is done to reduce the size of the input feature map to produce more robust features. Down-sampling helps the model to eventually get to a vector containing class scores at the end of the convolutional network. The final fully connected layers are responsible for calculating the score for each class and generating the output. Each neuron in these layers is connected to all the neurons from the previous map. The fully connected layers are usually adjusted for different vision tasks.

3.2 Vehicle Classification in Video Using the YOLOv3 Deep Learning Method

We apply the YOLOv3 deep learning method for vehicle classification in the video. Note that we have initially applied the full YOLOv3 for vehicle classification in the video, but it was not able to achieve real-time performance. To improve the computational efficiency, we apply the ”tiny” configuration which uses 13 convolutional layers instead of 75 layers, and six max-pool layers. In particular, the input image is first divided into an $S \times S$ grid. Note that each cell in the grid is used to detect only one object (in case there exists one) and the center of the object falls in that cell.

Fig. 4 shows the idea of our vehicle classification in a video using the YOLOv3 deep learning method. Note that among the many bounding boxes generated in one single network, the ones with the highest scores are chosen as the final detections [20]. In each cell, a fixed number (B) of bounding boxes with their confidence scores is generated. The confidence scores are calculated by multiplying the probability of each object and their intersection over union of the predicted box and the ground truth box. Each bounding box is indicated by five numbers: a quadruple (x, y, w, h), and the confidence score of the box. X and Y are the coordinates of the center of the box, and w and h are the width and height of the box respectively. These four numbers are float values relative to the absolute width and height of the image, and they can be somewhere between 0.0 and 1.0. The confidence score indicates the likeliness of the box containing an object. Each grid cell contains conditional class probabilities for the number of different classes, and therefore, for each category of objects, there is one probability in each cell, regardless of the value of B. Note that the conditional class probability means that the probability of the object belonging to a specific class is conditioned on the box containing an object. Thus, for each grid cell, there are $B \times 5$ numbers indicating the bounding box information and the C class probabilities. This prediction information is encoded as a tensor in the shape of $(S, S, B \times 5 + C)$.

Like R-CNN, the non-maximum suppression algorithm is used in YOLO to ignore the repetitive bounding boxes around the same object and to consider the box with the highest score value.

We use eight NJDOT traffic videos for evaluating both the Faster R-CNN deep learning vehicle classification method and the YOLOv3 deep learning vehicle classification method, which are trained using the three training data sets, respectively. In the process of evaluation, we consider two types of false positive (detecting background as a vehicle and misclassification) along with one false negative (missing data in the region of interest). We also count the ground truth or the unique counting to derive the success rate. We use the following formulas for calculating the success rate of the classification:

$$ER_{(Error\ rate)} = 100 \times \frac{FP_1 + FP_2 + FN}{GT_{(ground\ truth)}} \tag{1}$$

$$Classification\ success = 100 - ER \tag{2}$$

4 Experiments

Experiments are implemented using eight NJDOT traffic videos to evaluate the Faster R-CNN deep learning vehicle classification method and the YOLOv3 deep

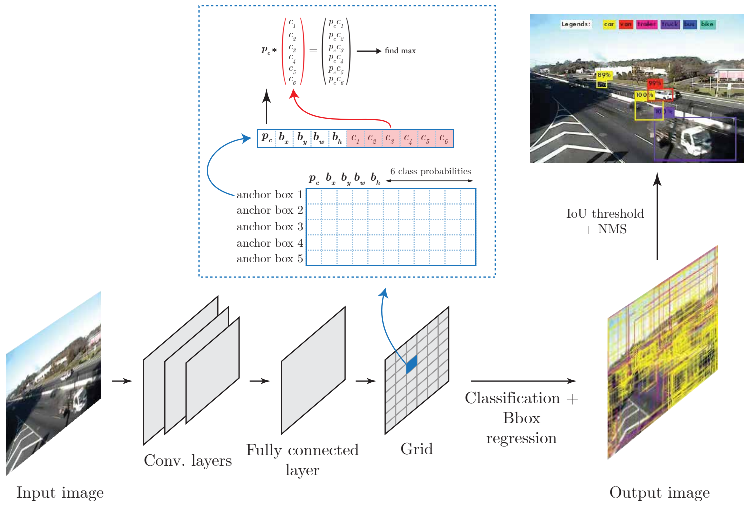


Fig. 4. Vehicle classification in the video using the YOLOv3 deep learning method. Among the many bounding boxes generated in one single network, the ones with the highest scores are chosen as the final detections.

learning vehicle classification method. We first create three training data sets using two videos, video 1 and video 4. Specifically, all the 13,500 frames of the first video, video 1, are used for defining the training data set 1. In each frame, the ground-truth bounding boxes are manually labeled. The training data set 2, however, contains only 1,227 frames by selecting only one frame from every 10 frames of video 4. These frames are further labeled manually for the vehicles and their corresponding class information. The training data set 3 includes all the 14,727 frames set 1 and set 2 to define a mixed training model. Table 1 summarizes the three training data sets.

Table 1. Three Training Data sets

	YOLO	Faster RCNN
Source (15fps, 352x240)	Reason for selection	Annotated images
Video 1	Uniformity of illumination	13500
Video 4	Illumination variance	1227
Videos 1 & 4	Mixture of data sets	14727

For training, we have used Nvidia GTX-745 GPU. In our test, YOLO has faster training and testing rate, unlike Faster R-CNN. For testing, Table 2 shows the average testing time for both methods. Note that each video is 15 minutes in length. The results in Table 2 indicate that YOLO is much faster than Faster R-CNN.

Table 2. Average testing time from eight videos corresponding to the three training data sets

Data Set 1		Data Set 2		Data Set 3	
YOLO	Faster R-CNN	YOLO	Faster R-CNN	YOLO	Faster R-CNN
6m2s	2h40m38s	6m4s	2h46m24s	6m5s	2h44m35s

Table 3. Detection result based on training samples from Video 1

Video	Unique counting	FP (BG as Vehicle)	FP (Mis-classification)	FN (Miss in ROI)	Classification Success	Errors
Detector →		YOLO				
Video 1	821	3	1	0	99.51%	0.49%
Video 2	881	2	23	3	96.82%	3.18%
Video 3	1057	15	23	0	96.40%	3.60%
Video 4	960	10	0	0	98.96%	1.04%
Video 5	1008	10	15	0	97.52%	2.48%
Video 6	988	9	24	0	96.66%	3.34%
Video 7	1017	6	10	0	98.43%	1.57%
Video 8	1148	5	26	1	97.21%	2.79%
Detector →		Faster RCNN				
Video 1	821	0	11	1	98.54%	1.46%
Video 2	881	1	22	2	97.16%	2.84%
Video 3	1057	2	35	8	95.74%	4.26%
Video 4	960	3	46	25	92.29%	7.71%
Video 5	1008	4	65	41	89.09%	10.91%
Video 6	988	3	49	43	90.38%	9.62%
Video 7	1017	2	54	52	89.38%	10.62%
Video 8	1148	3	74	44	89.46%	10.54%

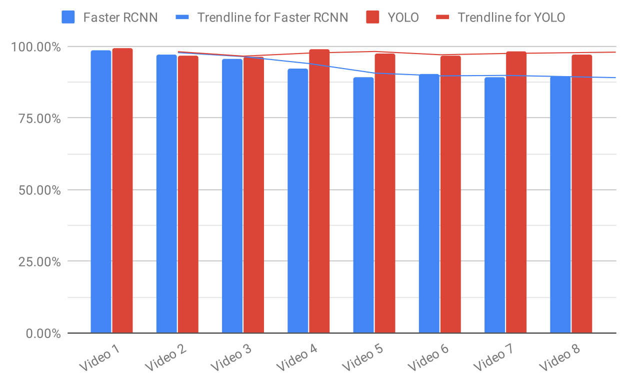


Fig. 5. Classification Success chart for YOLO and Faster R-CNN based on video 1 data

The training data set 1 from video 1 contains almost uniform illumination. The results from Table 3 and Fig. 5 show that YOLO generalizes well across all the eight videos and it performs better than the Faster R-CNN.

Table 4. Detection result based on training samples from Video 4

Video	Unique counting	FP (BG as Vehicle)	FP (Mis-classification)	FN (Miss in ROI)	Classification Success	Errors
Detector →		YOLO				
Video 1	821	8	34	236	66.14%	33.86%
Video 2	881	2	60	73	84.68%	15.32%
Video 3	1057	1	28	181	80.13%	19.87%
Video 4	960	0	5	0	99.48%	0.52%
Video 5	1008	1	9	0	99.01%	0.99%
Video 6	988	2	48	1	94.84%	5.16%
Video 7	1017	3	11	0	98.62%	1.38%
Video 8	1148	10	75	2	92.42%	7.58%
Detector →		Faster RCNN				
Video 1	821	5	53	16	90.99%	9.01%
Video 2	881	2	55	20	91.26%	8.74%
Video 3	1057	1	80	31	89.40%	10.60%
Video 4	960	0	15	3	98.13%	1.88%
Video 5	1008	1	44	10	94.54%	5.46%
Video 6	988	2	73	18	90.59%	9.41%
Video 7	1017	2	64	14	92.13%	7.87%
Video 8	1148	1	98	36	88.24%	11.76%

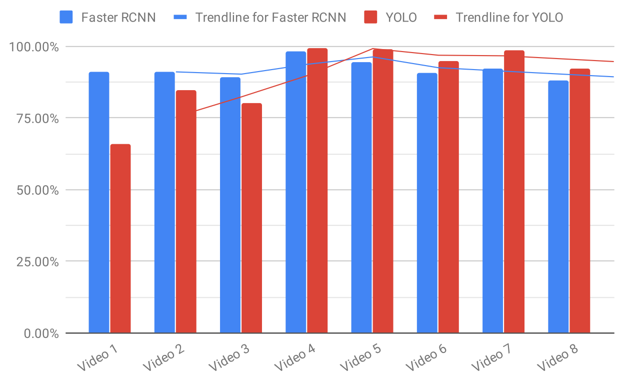


Fig. 6. Classification Success chart for YOLO and Faster R-CNN based on video 4 data

The training data set 2 from video 4 contains some illumination differences such as shadows and sun lights. Table 4 and Fig. 6 show the testing results using the eight videos. Note that for video 1 to video 8, the illumination differences increase. As a result, Fig 6 shows that both YOLO and Faster R-CNN display deteriorate generalization performance.

Table 5. Detection result based on training samples from Video 1 & 4

Video	Unique counting	FP (BG as Vehicle)	FP (Mis-classification)	FN (Miss in ROI)	Classification Success	Errors
Detector →		YOLO				
Video 1	821	0	2	0	99.76%	0.24%
Video 2	881	0	35	0	96.03%	3.97%
Video 3	1057	0	24	0	97.73%	2.27%
Video 4	960	0	10	0	98.96%	1.04%
Video 5	1008	0	19	0	98.12%	1.88%
Video 6	988	0	37	1	96.15%	3.85%
Video 7	1017	0	21	0	97.94%	2.06%
Video 8	1148	0	34	3	96.78%	3.22%
Detector →		Faster RCNN				
Video 1	821	0	13	4	97.93%	2.07%
Video 2	881	4	16	1	97.62%	2.38%
Video 3	1057	1	17	1	98.20%	1.80%
Video 4	960	0	6	0	99.38%	0.63%
Video 5	1008	2	21	2	97.52%	2.48%
Video 6	988	2	28	4	96.56%	3.44%
Video 7	1017	3	29	5	96.36%	3.64%
Video 8	1148	4	41	2	95.91%	4.09%

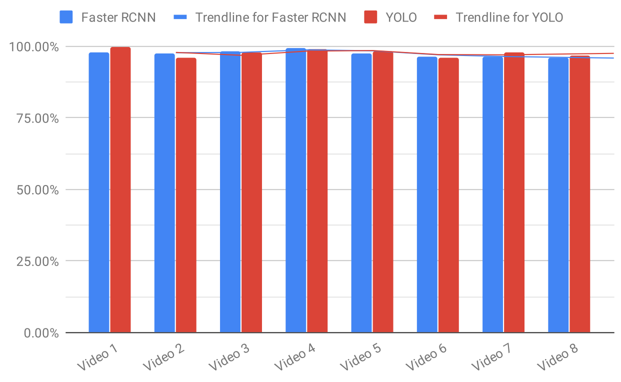


Fig. 7. Classification Success chart for YOLO and Faster R-CNN based on video 1, 4 data

The training data set 3 is a mixed data set containing frames from both video 1 and video 4. Table 5 and Fig. 7 show that both YOLO and Faster R-CNN achieve better generalization performance, and YOLO achieves better classification success rates than Faster R-CNN.

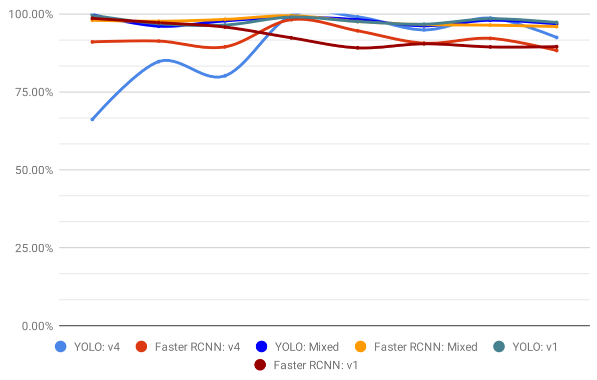


Fig. 8. Comparison of classification success chart for various data-sets

Fig. 8 illustrates the classification success rate using various training data sets. This figure essentially suggests that the training models with mixed images captured at different times of day with illumination variances help improve the generalization performance of the YOLO and the Faster R-CNN deep learning methods.

Fig. 9 shows some example testing results of the YOLO and the Faster R-CNN deep learning methods using the NJDOT traffic videos that are not seen

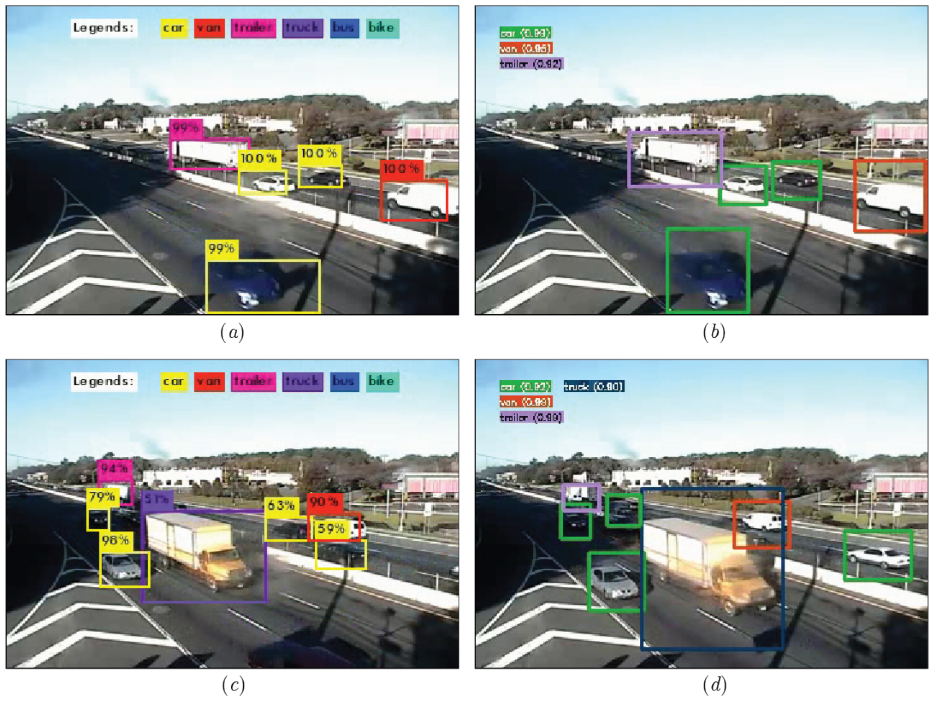


Fig. 9. Example testing results of the YOLO and the Faster R-CNN deep learning methods using the NJDOT traffic videos that are not seen during training. The frames in (a) and (c) show the vehicle classification results by using the YOLOv3 deep learning method. The frames in (b) and (d) show the vehicle classification results by using the Faster R-CNN deep learning method.

during training. Specifically, Fig. 9 (a) and (c) show the vehicle classification results by using the YOLOv3 deep learning method, and Fig. 9 (b) and (d) show the vehicle classification results by using the Faster R-CNN deep learning method. Fig. 9 reveals that the vehicles in all the frames are correctly classified without any false classification, and the YOLO method detects the vehicles more precisely (in tighter bounding boxes) than the Faster R-CNN method.

Through our research, we find that the deep learning models need a considerable amount of data and computational resources. The process of annotating and preparing necessary data to train a deep learning model is time-consuming. Besides, the complexity of multiple hidden layers in a deep network makes the interpretation and parameter configuration difficult. One of the main observations in this research is the limited ability of the deep learning methods in generalization. When a deep neural network is trained on specific data, which is gathered by intensive efforts, it will perform well on the same data in testing. However, when facing new slightly different data, even if it resembles the training data, the performance drops considerably. This is mainly because of the lack of reasoning and understanding the data, and pure dependence on experience and a large number of iterations. In the case of unfamiliar situations, the performance of deep learning models is not comparable to human performance. Observations from this research show that even the slight alternations in the visual situation, e.g., the presence of shadow or change of size and resolution makes the model less accurate. If the same model is used to detect vehicles in a more different situation, the results are expected to be even less accurate. For instance, changes in the angle of the camera overlooking the highway, lightening in different times of a day, and weather situations can result in more false positives, false negatives, and misclassifications.

YOLO is much faster in comparison with Faster R-CNN, and in case of their use for object detection and classification in video data, YOLO can process frames about 30 times quicker than Faster R-CNN. This property makes YOLO applicable for real-time detection in a video. The end-to-end training in a single network also improves the accuracy of this method. Another benefit of YOLO over methods based on region proposals is the fewer number of false positives, which is to detect a part of the background as an object. This is because YOLO sees the entire image as a whole and therefore, has a better outlook on the image. Newer versions of YOLO like YOLO9000 and YOLOv3 have improved the accuracy and mostly the speed of the original approach. The improvements are the cause of some modifications like training the model on multiple datasets and at multiple scales, using anchors to perform classification per anchor box instead of each grid cell, and pre-training on ImageNet at multiple scales. In YOLOv2, after training a classifier like VGG16, the YOLOv3 has improved in terms of accuracy by increasing the complexity of the architecture of Darknet, which YOLO is based.

5 Conclusions

Two representative deep learning methods, the Faster R-CNN and the YOLO, are applied for vehicle classification in videos, which has broad applications in intelligent transportation and smart cities. According to the Federal Highway Association (FHWA) vehicle types, six vehicle classes are defined: bike, car, truck, van, bus, and trailer. The training data sets are manually created from two New Jersey Department of Transportation (NJDOT) traffic videos and three training data sets are formed from video1, video 4, and both videos, respectively. New NJDOT traffic videos that are not seen during the deep learning network training are used to evaluate the vehicle classification performance in video. Using different training data sets, the Faster R-CNN and the YOLO deep learning methods display different performance in terms of the training time, the testing time, the vehicle classification accuracy, and the generalization performance. The experiments show the feasibility of vehicle classification in videos using deep learning methods, and reveal that the YOLO deep learning method is much faster that the Faster R-CNN deep learning method. Some limitations of the deep learning methods, such as the generalization performance, are discussed in the paper as well.

Acknowledgments: This work is partially supported by the NSF grant 1647170.

References

1. Bengio, Y., et al.: Learning deep architectures for ai. Foundations and trends® in Machine Learning **2**(1), 1–127 (2009)
2. Chen, S., Liu, C.: Clustering-based discriminant analysis for eye detection. IEEE Transactions on Image Processing **23**(4), 1629–1638 (2014)
3. Chen, S., Liu, C.: Eye detection using discriminatory haar features and a new efficient svm. Image and Vision Computing **33**, 68–77 (2015)
4. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on. vol. 1, pp. 886–893. IEEE (2005)
5. Deng, J., Berg, A., Satheesh, S., Su, H., Khosla, A., Fei-Fei, L.: Imagenet large scale visual recognition competition 2012 (ilsvrc2012). See net. org/challenges/LSVRC (2012)
6. Everingham, M., Eslami, S.A., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes challenge: A retrospective. International journal of computer vision **111**(1), 98–136 (2015)
7. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. International journal of computer vision **88**(2), 303–338 (2010)
8. Girshick, R.B.: Fast R-CNN. CoRR **abs/1504.08083** (2015), http://arxiv.org/abs/1504.08083v1
9. Gu, J., Liu, C.: Feature local binary patterns with application to eye detection. Neurocomputing **113**, 138–152 (2013)
10. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems. pp. 1097–1105 (2012)
11. Lin, T., Goyal, P., Girshick, R.B., He, K., Dollár, P.: Focal loss for dense object detection. CoRR **abs/1708.02002** (2017)
12. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: European conference on computer vision. pp. 740–755. Springer (2014)
13. Liu, C.: A bayesian discriminating features method for face detection. IEEE transactions on pattern analysis and machine intelligence **25**(6), 725–740 (2003)
14. Liu, L., Ouyang, W., Wang, X., Fieguth, P., Chen, J., Liu, X., Pietikäinen, M.: Deep learning for generic object detection: A survey. arXiv preprint arXiv:1809.02165 (2018)
15. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: Ssd: Single shot multibox detector. In: European conference on computer vision. pp. 21–37. Springer (2016)
16. Lowe, D.G.: Object recognition from local scale-invariant features. In: Computer vision, 1999. The proceedings of the seventh IEEE international conference on. vol. 2, pp. 1150–1157. Ieee (1999)

17. Nair, V., Hinton, G.E.: Rectified linear units improve restricted boltzmann machines. In: Proceedings of the 27th international conference on machine learning (ICML-10). pp. 807–814 (2010)
18. Ojala, T., Pietikainen, M., Maenpaa, T.: Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. IEEE Transactions on pattern analysis and machine intelligence **24**(7), 971–987 (2002)
19. Osuna, E., Freund, R., Girosit, F.: Training support vector machines: an application to face detection. In: Computer vision and pattern recognition, 1997. Proceedings., 1997 IEEE computer society conference on. pp. 130–136. IEEE (1997)
20. 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 (2016)
21. Redmon, J., Farhadi, A.: Yolo9000: better, faster, stronger. arXiv preprint (2017)
22. Redmon, J., Farhadi, A.: Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767 (2018)
23. Ren, S., He, K., Girshick, R.B., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. CoRR **abs/1506.01497** (2015), http://arxiv.org/abs/1506.01497v1
24. Rowley, H.A., Baluja, S., Kanade, T.: Neural network-based face detection. IEEE Transactions on pattern analysis and machine intelligence **20**(1), 23–38 (1998)
25. Uijlings, J.R., Van De Sande, K.E., Gevers, T., Smeulders, A.W.: Selective search for object recognition. International journal of computer vision **104**(2), 154–171 (2013)
26. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on. vol. 1, pp. I–I. IEEE (2001)