Vulnerable Road Users Detection based on Convolutional Neural Networks

Abdelhamid Mammeri, Abdul Jabbar Siddiqui, Yiheng Zhao, Barry Pekilis
National Research Council
Ottawa, Canada
{Abdelhamid.Mammeri, AbdulJabbar.Siddiqui, Yiheng.Zhao, Barry.Pekilis}@nrc-cnrc.gc.ca

***Abstract*—The roads support many different types of users. With geared efforts to advance connected and autonomous vehicles (CAVs), smart mobility systems, and advanced driver assistance-based vehicles, the safety of road users becomes a growing concern. Some users may require more cautious interactions and support to ensure safe usage of the road infrastructures. While considerable effort has been done to detect different types of road users or objects from a vehicle's viewpoint, there are certain classes of vulnerable road users which have been overlooked in prior works. The objective of this work is to detect vulnerable road users (e.g., Strollers, Motorbikes, and Bicycles) in order to aid in reduction of collisions. We investigate the performance of one-stage and two-stage deep object detection methods in detection of said vulnerable users. Since there is a lack of publicly accessible datasets containing objects of our interest from an infrastructure viewpoint, we introduce our own dataset collected from a road side. We highlight the benefits and shortcomings of the studied methods in the context of vulnerable road users detection under challenging conditions such as occlusions.**

***Index Terms*—intelligent transportation systems, deep learning, vulnerable road users, transportation safety**

I. Introduction

The key enablers of safe connected and autonomous vehicles (CAVs) and intelligent transportation systems (ITS) include object detection and classification methods. Although the problem of object detection and classification is not new to the CAV/ITS community, prior works have overlooked certain categories of objects (i.e., vulnerable road users). Majority of the publicly available datasets have ignored these classes, with little or no samples at all. One such important class of vulnerable road users is that of Strollers.

In order to achieve a fully safe intelligent transportation system, prevention of accidents and fatalities (especially for vulnerable road users) is of utmost importance. According to a recent report by Statistics Canada [1], around 74 cyclists died in Canada, based on data from the Canadian Vital Statistics: Death Database (CVSD) over years 2006 to 2017. Of the fatal cycling events over these years, 73% were due to collisions with a motor vehicle, 25% were due collisions with another object or cyclist, and the cause behind the remaining 2% was unknown. Moreover, in a report by NHTSA's National Center for Statistics and Analysis [2], from 2008 to 2017, there were around 1420 fatalities involving persons on strollers and other personal conveyances.

The need to detect and identify vulnerable road users such as Strollers arises from the fact that they require more cautious interactions and behaviors by other road users around them. Enabling the infrastructure to detect and identify such users could aid in many safety applications. For example, road side units could inform the CAVs approaching the area about the presence of such vulnerable road users and their requirements, warning the respective CAVs to be more careful.

The problem of detecting objects like strollers from footage taken by camera(s) mounted on road side unit(s) poses certain challenges for object detection algorithms. First is due to the variation in designs, shapes, and styles of these objects. Second is due to the occurrence of occlusions due to other road users, and in the case of strollers, occlusion by the person pushing the stroller. Third is related to the size of these objects in the images: since the camera is mounted on a road side unit and at a certain height, the objects' sizes may be very small as compared to objects sizes in images captured by vehicle-mounted cameras. Fourth arises out of weather and other related environmental factors such as fog, rain, snow, bright sunlight, shadows and darkness.

In this work, we report the following contributions. We investigate the use of convolutional neural networks-based methods to detect vulnerable road users such as strollers, motorbikes and bicycles, from a road infrastructure point of view. Specifically, we study two classes of CNN-based methods: (i) Single Stage, and (ii) Two-Stage methods. The former class includes methods like YOLOv3 and its variants whereas the latter class includes methods like Faster RCNN. For experimental evaluations, We introduce a new dataset (collected from a real roadside infrastructure) containing vulnerable road users such as strollers, bicycles and motorbikes. We analyse the performance of the said methods and discuss their pros or cons.

The remainder of the paper is organised as follows. In Section II, we describe the related works, the selected CNN-based methods studied in this work, and some of the popular object detection datasets. In Section III, we provide a description of our dataset and discuss its advantages as compared to prior datasets. Section IV then provides the experimental results and discussions on the performance of selected methods in detection of strollers, motorbikes and bicycles. The paper finally concludes and presents some points of future work in Section V.

978-1-7281-5628-6/20/$31.00 ©2020 Crown

II. Related Work

Object detection is one of the most common computer vision tasks focused on locating and identifying objects of certain classes in the image or video. Many applications involve the object detection task, e.g., ITS (Intelligent Transportation Systems) and ADAS (Advanced Driver-Assistance Systems). In such applications, detection of vehicles, drivers, and vulnerable road users is one of the foundations that help to indicate the real situation, surroundings, and environment.

Implementing object detection can be achieved in various ways, based on either traditional frameworks or deep machine learning models. Viola-Jones framework [3] is one of the classical traditional detection algorithm for face detection. First, Haar-like algorithm provides features from integral images. Then, AdaBoost algorithm is applied to select the most useful features. Finally, a cascade classifier is applied to recognize faces. Although machine learning techniques are used, the performance is not always desirable in some conditions, limited by the number of feature types.

Relatively, deep machine learning based methodologies are recently introduced to further improve the detection accuracy. CNNs (Convolutional Neural Networks) are commonly used for learning feature kernels from training dataset, which include much more feature types than in Haar-like algorithms. As a consequence, CNN models are powerful to describe objects' properties without much manual designing of kernel matrices. Besides, the training process is dramatically simplified with the help of deep machine learning packages (e.g., Tensorflow, Pytorch, etc.). Based on whether ROIs (Regions of Interest) are extracted independently, deep machine learning-based methods can be categorized into two: two-stage and one-stage models.

In this section, we first describe the two-stage and one-stage object detection methods based on CNNs. Then, we briefly describe some of the most popular datasets used by object detection works and highlight the lack of attention to certain vulnerable road users (VRUs).

A. Two-Stage Methods

The two-stage object detection methods mainly comprise of the following stages: (i) a region proposal stage, and (ii) classification and bounding box regression stage. The first stage produces a prediction of potential candidate regions that are expected to contain the objects of interest. The second stage then classifies what object is contained in the selected regions as well as refines the bounding box around those objects.

In this work, we study the popular two-stage method known as Faster RCNN [4]. Below, we shall give an overview of the method and its architecture. Faster RCNN comprises of three networks: (i) Backbone, (ii) Region Proposal Network (RPN) and (ii) Detector Network. The backbone network is a CNN that learns to extract useful features from the input images. The RPN benefits in terms of time cost by sharing computations with the object detection network, leveraging the same backbone. To generate the region proposals, RPN looks at the image features through a certain number of anchors of certain different shapes and sizes. The RPN learns to classify the anchor regions, predicting whether these contain objects of interest or not. In addition, the RPN refines the anchors' coordinates to yield refined bounding boxes. Only the anchors that most likely contain an object of interest are retained for further steps, yielding the candidate ROIs. These ROIs are filtered through non-maximum suppression and the final list of region proposals is fed into ROI Pooling.

Since the ROIs resulting from the RPN could be of various sizes, the ROI Pooling module is used to transform all the ROIs to a certain size. Each of the ROIs is roughly-equally split into a fixed number of sub-regions, say k. Then, on each such sub-region, Max Pooling operations are performed. In this manner, ROI Pooling produces transformed feature maps of size k, irrespective of their original sizes. These transformed and same size ROIs feature maps are then used to feed to final classifier and regressor. While the classifier learns to predict the class of the object contained in the ROI, the regressor learns to refine the bounding box around the object. Further details can be found in [4]. In this work, we study the Faster RCNN model with ResNet50-FPN as a backbone which utilizes ideas from ResNet [5] and Feature Pyramid Network [6].

B. One-Stage Methods

Compared to two-stage methods, independent ROI extractors (e.g., selective search and RPN) are not necessary in one-stage methods. With the fixed number, predictions are calculated directly from a dense image over all pre-defined grids. As a consequence, the detectors tend to be much faster than two-stage models. YOLO [7] is the pioneer in this domain, followed by SSD [8], YOLO9000 [9], YOLOv3 [10], CornerNet [11], and RetinaNet [12].

We choose the third version of YOLO model (named YOLOv3) for the experiments due to its high accuracy and fast speed. Many improvements are applied on YOLOv3 in [10]: (1) authors designed a deeper backbone CNN model named Darknet53 to improve the quality of feature maps; (2) the output feature maps are the concatenations from three sizes of intermediate $(52 \times 52, 26 \times 26)$ and last feature maps (13×13) with separate refining layers. By doing so, the accuracy of small object detection is improved; (3) authors also introduce anchors into their model. The dimensions of the feature maps from each scale are $S \times S \times A \times (5+C)$. S is the size of feature map. A is the number of pre-defined anchors. 5 includes information of bounding boxes (x, y, w, h), and confidence score to decide objectiveness. C refers to the number of classes. 4) in terms of loss function, authors replaced the last three terms of YOLO9000 by cross-entropy error terms so that the prediction of confidence and class predictions are predicted through logistic regression. 5) softmax is removed from the model to eliminate the dependency of output parameters.

We study the YOLO framework with two types of backbone networks: 1) Darknet53, and 2) ResNet50 with DLA (deep layer aggregation). DLA [13] is a hierarchical structure that can be appended to most of the CNN models. Authors

TABLE I
RELATED DATASETS FOR VULNERABLE ROAD USERS DETECTION

Ref.	Name	Strollers	Bicycles	Motorbikes
[15]	Motorbike Dataset	×	×	✓
[16]	MIO-TCD	×	✓	✓
[17]	Berkeley Deep Drive (BDD100K)	×	✓	✓
[18]	Open Images Dataset	×	✓	✓
[19]	COCO	×	✓	✓
[20]	PACAL VOC	×	✓	✓
[14]	nuScenes	✓	✓	✓
This	VRU Dataset v1	✓	✓	✓

augmented standard architectures with deeper aggregation of layers to better fuse information across layers. However, in their paper, no experiments are implemented for object detection task. In this work, we evaluate the performance of DLA on detection task.

C. Datasets

Over the recent years, many datasets have been collected for the purposes of road object detection, e.g., ImageNet, PASCAL VOC, etc. These could be broadly grouped into two categories: (i) Vehicle-mounted camera-based, and (ii) Roadside fixed camera-based. The datasets belonging to the former category are collected through cameras installed on moving vehicles whereas datasets of the latter category are collected by cameras fixed at the roadsides. In this work, we focus on the latter category. Most of the datasets of the latter category do not contain samples representing vulnerable road users such as strollers. Despite the advances in object detection in the recent years, this object class has been long ignored.

In Table I we list some relevant and popularly used object detection datasets relevant to the cause of vulnerable road users detection. One can see how current datasets lack in representing certain important types of vulnerable road users such as strollers. Although one of these datasets (nuScenes [14]) does contain samples for strollers, it was collected by a camera mounted on a car driving around and hence doesn't represent the diversity of viewpoints as would occur in images captured by cameras on a roadside infrastructure. Table I is not a comprehensive list by any means, and as work in the area evolves, more datasets may be produced. Our dataset is the first of its kind, from a roadside infrastructure perspective, in introducing attention towards certain classes of vulnerable road users which have been neglected in popular works. We continue to expand our dataset adding more such classes, and plan to publish it in a future work. We hope our work will motivate further research into the problem of VRU detection.

III. PROPOSED DATASET

We now introduce our *Vulnerable Road Users Dataset*, examples of which are shown in Figure 1. The VRU dataset was collected from a camera fixed on a roadside infrastructure in a Canadian city. The area was selected based on observation of diverse classes of road users passing through. As such, for this work, we focused on the following classes of vulnerable road users, hereby referred to as objects of interest (OIs): (1) Motorbike, (2) Bicycle, and (3) Stroller.

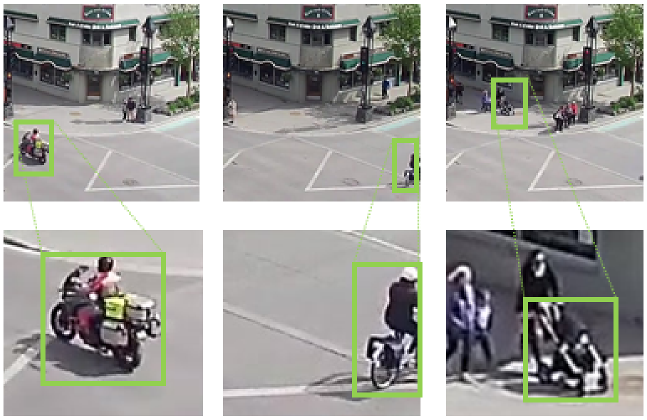


Fig. 1. Examples from our Vulnerable Road Users Dataset for the objects of interest: [Left-Right] Motorbike, Bicycle, and Stroller.

The videos were recorded over a couple of hours during the day. From these videos, we extracted the sequences where OIs were seen, and ignored the remaining parts of the videos. From these sequences, we extracted frames (at a rate of 1fps) to be manually labelled. The OIs in the selected frames were annotated using the LabelImg [21] tool. During annotation, we observed many cases where the OIs were occluded. In such cases, we labelled the instance if the OI (or its part, e.g. wheel of a stroller) was partially yet identifiably visible to the annotator. It is worth noting here that such cases add to the challenges in achieving robust detection systems. Another challenge posed by the dataset includes the different views of OIs which may pass by the camera at different angles.

Since we focus on the mentioned OIs which occur less frequently than other objects, we adopted data augmentation techniques to increase the number of samples for training and evaluation. We apply the following augmentation transformations, with varying parameters, resulting in 30 different images for each frame: (i) Horizontal Flipping, (ii) Scaling, (iii) Rotation, (iv) Random HSV Modification (to change hue, saturation and brightness). First, a frame is horizontally flipped. On the original image and its flipped version, five different scaling transformations, 5 different rotation operations, and 5 different random HSV modifications are performed, resulting in 30 augmented images for each original image.

In the scaling and rotation transformations, OIs whose bounding boxes have an area of less than 25% (after the transformations) are ignored. In both these types of transformations the image resolution is not altered, and hence any area left empty after the transformation is filled with black color.

We randomly split the dataset into training, validation and testing sets following a ratio of 60%-20%-20% respectively. Table II shows the number of samples of each OI class. The total number of OI samples in the dataset is 12839.

To study the effectiveness of detection methods for different sizes of objects' appearance in the images, we group the object

TABLE II
VRU DATASET COMPOSITION

OI Class	Training	Validation	Testing	Total
Motorbike	78	17	29	124
Bicycle	3673	1194	1228	6095
Stroller	3885	1422	1313	6620
Total # OI samples				12839

TABLE III
VRU DATASET DISTRIBUTION (BASED ON AREA OF OI BOUNDING BOXES)

OI Class	Small	Medium	Large	Total
Motorbike	31	0	93	124
Bicycle	2862	2622	611	6095
Stroller	5607	959	54	6620
Total	8500	3581	758	12839

instances in the dataset into three categories based on area: (i) small: area is less than 1502, (ii) medium: area is greater than or equal to 1502 but less than 2630, (iii) large: area is greater than or equal to 2630. These ranges were selected based on a KMeans clustering of the bounding boxes' areas in the VRU dataset. The number of object instances in each of the three area-based categories are: $8500, 3581, 758$, respectively. In the current version of the dataset, no instances of motorbikes fall into the medium area category. In future, we shall expand the dataset collecting more samples for each area category. Table III provides the number of object instances of each class and area category.

IV. EXPERIMENTS AND DISCUSSIONS

We evaluate the selected methods using the proposed VRU Dataset based on the performance metrics defined below. Further, we describe the training and validation procedure for the three models and present the testing results and discussions.

A. Performance Metrics

1) Intersection over Union (IoU): In object detection works, the task of the predictor is not only to identify the class of the object, but also to localize the object in the image by predicting a tightly fitting bounding box. To measure the performance in detection and localization of objects of interest, an important factor to consider is the Intersection over Union (IoU) scores between the predicted bounding boxes and the ground truth bounding boxes. In brief, it is defined as the ratio of area of overlap to the area of union between a predicted bounding box and a ground truth bounding box. So, given a predicted bounding box PBB_i and corresponding ground truth bounding box GTB_k, IoU is defined as:

$$IoU(PBB_i, GTB_k) = \frac{Area(PBB_i) \cap Area(GTB_k)}{Area(PBB_i) \cup Area(GTB_k)} \quad (1)$$

An acceptable PBB_i would be one that satisfies a given IoU threshold, IoU_{th} which usually ranges between 0.5 to 0.95 as per the COCO evaluation standard [19].

2) Average Precision: In general, precision, with respect to a certain class, is defined as the ratio of number of true positives to the total number of predictions for that class. And average precision (AP) is calculated as per the COCO method [19] (i.e., using 101 recall points). Since the precision of a method depends on the IoU_{th}, one way of averaging the precision scores is across a range of IoU_{th}. For example $AP_{0.5-0.95}$ refers to average precision scores across a range of IoU_{th} from 0.5 to 0.95, in steps of say 0.05. Similarly, $AP_{0.5}$ is the AP at $IoU_{th} = 0.5$. In order to quantify the overall performance of a method, a metric known as mean average precision (mAP) is calculated by averaging the classwise $AP_{IoU_{th}}$ scores (for the given IoU_{th} and N classes). So, $mAP_{IoU_{th}} = \sum_{c=1}^{N} AP_{IoU_{th}}^{c}/N$

3) Average Recall: While precision measures how many of the predictions made are actually correct, recall on the other hand measures how many of the ground truth samples were correctly detected and localized. So, in general, recall is the ratio of true positives to the total number of ground truth samples for a certain class. The average recall (AR) metric is used to to present the performance across all classes or a range of IoU_{th}'s. The mAR metric, defined similar to the mAP of COCO method [19], is the average AR scores of all classes for a given (range of) IoU_{th}.

4) Testing Execution Time: Besides evaluating the performance based on mAP and mAR, we measure the execution time in testing phase for the three studied models. The execution time is an important factor to consider as different applications may have different latency requirements.

5) Confusion Matrix: Depicts the percentage of samples of each ground-truth class (represented by columns) classified to each class (represented by rows). the main diagonal values are the True Positives, while the other values indicate the False Positives or False Negatives. In a column i, the value at (i, i) is the true positives count (TP_i) for class-i; the values in (j, i), where $j \neq i$, show the False Negatives (FN_i), i.e., the number of samples of class-i that were mis-classified to class-j. Moreover, for a class-j, the values (j, i) in column-i, $i \neq j$, are the counts of FP_j.

B. Training and Validation

In this section, we describe the training and validation of the three methods studied in this work for detection of strollers, motorbikes and bicycles, in terms of their mAP and mAR scores over 100 epochs. As shown in Figure 2, Faster RCNN yielded higher mAP and mAR compared to YOLOv3 methods. A reason could be that Faster RCNN was trained starting from ImageNet-based pre-trained model weights.

As for YOLOv3 (with Darknet53) and YOLOv3 (with DLA102), one can observe that the models progress achieving higher mAPs and mARs with more number of epochs, although relatively less stably than the Faster RCNN. This can be attributed to the fact that we trained both YOLOv3 models from scratch on our dataset. Due to the unavailability of pre-trained weights for DLA102 backbone network, in this work, we train YOLOv3 (DLA102) from scratch. In order to have

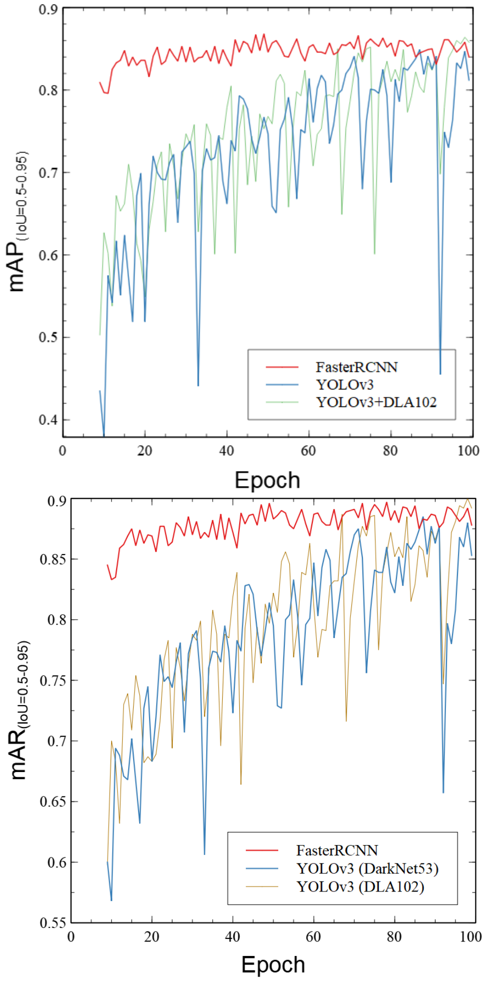


Fig. 2. Comparing the mAP and mAR scores (with $IoU = 0.5$ to 0.95) of Faster RCNN (ResNet50-RPN), YOLOv3 (Darknet53) and YOLOv3 (DLA102) methods. (Figure best viewed in color)

a fair comparison, hence, YOLOv3 with Darknet53 backbone was also trained from scratch.

In each epoch, the trained model of each method is evaluated on the validation dataset. Based on performance scores (AP and AR), the best epoch's model is selected as the final model to be used for evaluations on the testing dataset. Specifically, given the following metrics: (a) $AP_{0.50-0.95}$, (b) $AP_{0.50}$, (c) $AP_{0.75}$, (d) $AR_{0.50}$, (e) $AP_{0.75}$, the selection criteria for the best model is based on three dictionary comparisons: Comp1, Comp2, Comp3. Comp1 selects the model which has highest values for (a). If there is more than candidate with highest value for (a), then compare based on (b), and so on in this order. In Comp2, we select the model which has the highest values for (b). If more than one model achieves the highest (b) value, then compare based on (a), then (c)-(e) in this order. In Comp3, we select the model which has the highest values for (c), and continue comparisons based on values for (a), (b), (d), (e), in this order. Finally, the model found with Comp1, Comp2 and Comp3 is selected as the final learnt model of the respective method. If more than one model satisfies Comp1, Comp2, Comp3, we select the one with the lower epoch index as a higher epoch index could suffer from overfitting. As long as the APs and ARs of the selected model are similar to those of the converged one, we can assume there is no underfitting.

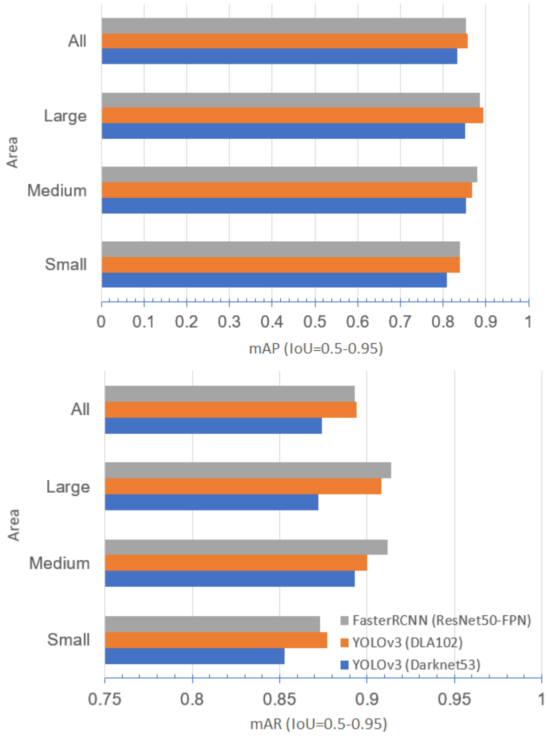
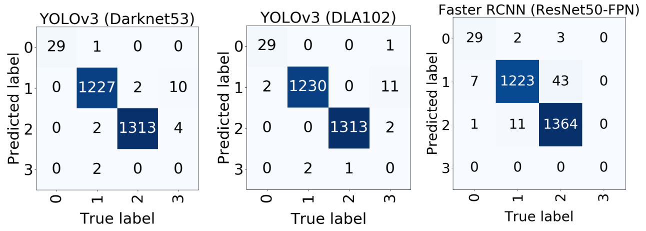


Fig. 3. mAP (top) and mAR (bottom) of Faster RCNN (ResNET50-FPN backbone), YOLOv3 (Darknet53 backbone) and YOLOv3 (DLA102 backbone) for object instances of small, medium, and large areas (in pixels). 'All' corresponds to mAP and mAR averaged over all object instances regardless of area.



YOLOv3 (Darknet53)

	0	1	2	3
0	29	1	0	0
1	0	1227	2	10
2	0	2	1313	4
3	0	2	0	0

YOLOv3 (DLA102)

	0	1	2	3
0	29	0	0	1
1	2	1230	0	11
2	0	0	1313	2
3	0	2	1	0

Faster RCNN (ResNet50-FPN)

	0	1	2	3
0	29	2	3	0
1	7	1223	43	0
2	1	11	1364	0
3	0	0	0	0

Fig. 4. Confusion Matrices of the studied methods in this work: YOLOv3 and Faster RCNN. Class indices 0,1,2,3 correspond to Motorbike, Bicycle, Stroller and Background..

In these comparisons, the order of parameters indicates the priority levels. For example, in Comp1, $AP_{0.5-0.95}$ has highest priority. As a consequence, Comp1 focuses on performance in terms of $AP_{0.5-0.95}$. Based on the above criteria, the selected models for Faster RCNN, YOLOv3 (Darknet53), and YOLOv3 (DLA102) were from Epochs $49, 87$ and 98, respectively.

C. Testing

In this section, we give the evaluation results of specified one stage detectors (i.e. YOLO-Darknet53, and YOLOv3 with ResNet-DLA as backbone network) and two-stage detector (Faster RCNN) in terms of precision, recall and time cost based on IOU thresholds of 0.5 and 0.75.

TABLE IV
TESTING TIME COMPARISONS)

Method	Testing Time (s)
YOLOv3 (Darknet53)	0.0156
YOLOv3 (DLA102)	0.0312
Faster RCNN (ResNet50-FPN)	0.1321

Looking at the mAP and mAR of the three methods, for the three area categories, we make the following observations (Figure 3). One can see that the performance of all three methods is lowest for the small objects which illustrates that small objects detection remains a pending challenge in object detection works. Furthermore, the performance of YOLOv3 is lowest amongst the three methods, for the small objects category. However, by adding the DLA102 backbone instead of the Darknet53, its performance in terms of mAP and mAR significantly improves by around 2.9% and 2% respectively. The performance of YOLOv3 with DLA102 backbone was in fact similar to that of Faster RCNN for the small objects. When comparing between the performance of YOLOv3 (Darknet53) and Faster RCNN, we find that the latter yields mAP and mAR of around 2.5 and 2.2 percentage points higher respectively. The overall mAP and mAR scores of Faster RCNN are very close to those of YOLOv3 (DLA102).

Considering the overall performance of the three methods, YOLOv3 (DLA102) yielded the best results in terms of mAP and mAR scores (averaged over all object instances of all area ranges), as shown in Figure 3. This validates the performance gains of incorporating DLA102's hierarchical deep layer aggregation into YOLOv3 backbone.

We also evaluate the testing time requirements of the three methods. In Table IV we provide the average testing time consumed by the model to make the predictions. YOLOv3 (Darknet53) requires the least execution time amongst the studied models owing to its simpler architecture compared to the other models. YOLOv3 (DLA102) and Faster RCNN consumed around 50% and 88.2% (respectively) more execution time than YOLOv3 (Darknet53), with higher mAP and mAR.

V. CONCLUSIONS AND FUTURE WORK

In this work, we presented the case of detecting vulnerable road users (VRUs) from a roadside perspective, focusing on certain classes of road users which have not been given attention in prior works (e.g., Strollers). We investigate the performance of three convolutional neural networks-based models for the task of detecting and localizing these VRUs in images collected from a real roadside. In future work, we plan to expand our dataset considering special classes of VRUs, different weather and illumination conditions, which present unique challenges for detection and localization methods.

REFERENCES

[1] "Circumstances surrounding cycling fatalities in canada, 2006 to 2017," July 2019, [Statistics Canada]. [Online]. Available: https://www150.statcan.gc.ca/n1/pub/82-625-x/2019001/article/00009-eng.htm

[2] "Traffic safety facts: 2017 data," March 2019, [NHTSA's National Center for Statistics and Analysis, U.S. Department of Transportation, National Highway Traffic Safety Administration (NHTSA), USA]. [Online]. Available: https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/812681

[3] P. Viola and M. Jones, "Robust real-time face detection," pp. 137–154, 2004.

[4] S. Ren, K. He, R. B. 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.

[5] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*. IEEE Computer Society, 2016, pp. 770–778.

[6] T. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie, "Feature pyramid networks for object detection," in *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*. IEEE Computer Society, 2017, pp. 936–944.

[7] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[8] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. E. Reed, C. Fu, and A. C. Berg, "SSD: single shot multibox detector," in *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part I*, ser. Lecture Notes in Computer Science, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds., vol. 9905. Springer, 2016, pp. 21–37.

[9] J. Redmon and A. Farhadi, "YOLO9000: better, faster, stronger," in *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*. IEEE Computer Society, 2017, pp. 6517–6525.

[10] ——, "Yolov3: An incremental improvement," *CoRR*, vol. abs/1804.02767, 2018. [Online]. Available: http://arxiv.org/abs/1804.02767

[11] H. Law and J. Deng, "Cornernet: Detecting objects as paired keypoints," in *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XIV*, ser. Lecture Notes in Computer Science, V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds., vol. 11218. Springer, 2018, pp. 765–781.

[12] T. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 2, pp. 318–327, 2020.

[13] F. Yu, D. Wang, E. Shelhamer, and T. Darrell, "Deep layer aggregation," in *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*. IEEE Computer Society, 2018, pp. 2403–2412.

[14] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," *arXiv preprint arXiv:1903.11027*, 2019.

[15] J. E. Espinosa, S. A. Velastin, and J. W. Branch, "Motorcycle detection and classification in urban Scenarios using a model based on Faster R-CNN," *arXiv:1808.02299 [cs]*, Aug. 2018, arXiv: 1808.02299.

[16] Z. Luo, F. Branchaud-Charron, C. Lemaire, J. Konrad, S. Li, A. Mishra, A. Achkar, J. Eichel, and P. Jodoin, "Mio-tcd: A new benchmark dataset for vehicle classification and localization," *IEEE Transactions on Image Processing*, vol. 27, no. 10, pp. 5129–5141, Oct 2018.

[17] F. Yu, W. Xian, Y. Chen, F. Liu, M. Liao, V. Madhavan, and T. Darrell, "Bdd100k: A diverse driving video database with scalable annotation tooling," 2018.

[18] A. Kuznetsova, H. Rom, N. Alldrin, J. Uijlings, I. Krasin, J. Pont-Tuset, S. Kamali, S. Popov, M. Malloci, T. Duerig, and V. Ferrari, "The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale," 2018.

[19] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *Computer Vision – ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham: Springer International Publishing, 2014, pp. 740–755.

[20] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes challenge: A retrospective," *International Journal of Computer Vision*, vol. 111, no. 1, pp. 98–136, Jan. 2015.

[21] Tzutalin, "Labelimg," [Git code 2015]. [Online]. Available: https://github.com/tzutalin/labelImg