Evaluation of Detection and Segmentation Tasks on Driving Datasets

Deepak Singh[1], Ameet Rahane[2], Ajoy Mondal[1],
Anbumani Subramanian[3], and C. V. Jawahar[1]

[1] International Institute of Information Technology, Hyderabad, India
[2] University of California, Berkeley,
[3] Intel, Bangalore, India,
{deepak.singh@research., ajoy.mondal@,jawahar@}iiit.ac.in
ameetrahane@berkeley.edu, anbumani.subramanian@intel.com

Abstract. Object detection, semantic segmentation, and instance segmentation form the bases for many computer vision tasks in autonomous driving. The complexity of these tasks increases as we shift from object detection to instance segmentation. The state-of-the-art models are evaluated on standard datasets such as PASCAL-VOC and MS-COCOC, which do not consider the dynamics of road scenes. Driving datasets such as Cityscapes and Berkeley Deep Drive (BDD) are captured in a structured environment with better road markings and fewer variations in the appearance of objects and background. However, the same does not hold for Indian roads. The Indian Driving Dataset (IDD) is captured in unstructured driving scenarios and is highly challenging for a model due to its diversity. This work presents a comprehensive evaluation of state-of-the-art models on object detection, semantic segmentation, and instance segmentation on-road scene datasets. We present our analyses and compare their quantitative and qualitative performance on structured driving datasets (Cityscapes and BDD) and the unstructured driving dataset (IDD); understanding the behavior on these datasets helps in addressing various practical issues and helps in creating real-life applications.

Keywords: Object detection, semantic segmentation, instance segmentation

1 Introduction

In computer vision, the granularity of the label increases as we move from object detection to instance segmentation. We perform classification and localization of the objects of interest in object detection, but in semantic segmentation, we also consider the boundary of each object during classification. Further in instance segmentation, we differentiate each instance of the object during segmentation. Figure 1 captures the increasing complexity in each of the tasks.

For autonomous driving applications, datasets like Cityscapes [8], BDD [34] and IDD [27] are collected in structured and unstructured driving conditions respectively. Academic datasets such as PASCAL-VOC [9] and MS-COCO [19] are

commonly used for benchmark object detection, semantic segmentation, and instance segmentation.

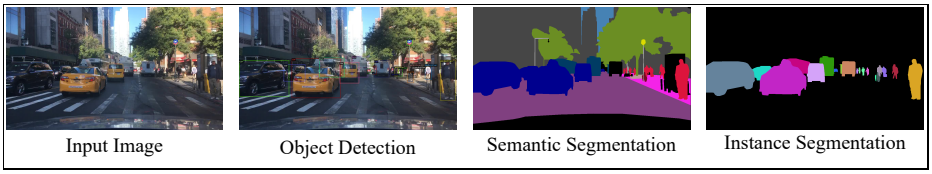

Fig. 1. Illustrates popular tasks of computer vision on road scenes. We can notice that the granularity of the label becomes more complex as we move from Object Detection to Instance Segmentation. (Best viewed in color and zoomed).

Our **contributions** in this work comprise of evaluation and analyses of various state-of-the-art deep learning models on object detection, semantic segmentation, and instance segmentation with structured datasets - Cityscapes [8] and BDD [34], and an unstructured driving dataset - IDD [27]. We evaluate the performances with:

(i) four object detectors: Faster R-CNN [25], SSD [20], RetinaNet [18], and YOLOv3 [24],
(ii) three semantic segmentation architectures: PSPNet [37], ERFNet [26], and DRN [35], and
(iii) three instance segmentation techniques — Mask R-CNN [12], Cascade Mask R-CNN [4], and Mask Scoring R-CNN [14].

To our knowledge, this is the first comprehensive work to use driving datasets instead of standard academic datasets such as PASCAL-VOC [9] and MS-COCO [19], to perform quantitative and qualitative analyses of various deep learning models on multiple tasks. Understanding the behavior of state-of-the-art object detection, semantic segmentation, and instance segmentation techniques on driving sequences play a vital role in creating real-life applications.

2 Related Work

Object Detection: Existing Deep Convolutional Neural Network (DCNN) based object detectors are of two categories: (i) two-stage detectors and (ii) one-stage detectors. The two-stage detectors comprises of a region-proposal step, region classification and regression step. Some popular works include [11, 10, 25], several modified architectures [12, 4] have also been developed to improve detection accuracy. Though two-stage detectors produce high accuracy, they cannot be used for real-time applications due to their high computation time. In contrast, one-stage detectors predict boxes from input images directly without a region proposal step and hence are time efficient, lending their use for real-time applications. The notable work of Redmon *et al.* in YOLO [22] laid the foundation

for several other versions such as YOLOv2 [23] and YOLOv3 [24]. Other popular one-stage object detectors are SSD [20], MT-DSSD [1], RetinaNet [18], M2Det [38], and RefineDet [36].

Semantic Segmentation: Deep Convolutional Neural Network (DCNN) based semantic segmentation techniques demonstrate improvements by replacing the fully-connected layer in image classification network with convolution layers, calling it Fully Convolutional Network (FCN) [21]. Several methods [6, 37] have been developed to overcome the limitations of FCN [21]. While methods [37, 7, 21] have been developed by combining multi-scale features to improve the segmentation performance, another approaches [6, 2] involve semantic segmentation based on structure prediction. Running DCNNs on mobile platforms (e.g., drones, robots, and smartphones) requires networks to work in real-time on embedded devices with space and memory constraints. Some lightweight networks in real-time semantic segmentation do exist [30, 26, 29] in the literature.

Instance Segmentation: Instance segmentation assigns different labels to each instance of an object belonging to the same category. Pose estimation, surveillance, robotics, and self-driving cars are areas where instance segmentation plays a key role. Instance segmentation techniques are of two categories: (i) two-stage, and (i) one-stage. Some of the latest works for two-stage approaches constitutes Mask R-CNN [12], Cascade Mask R-CNN [4], Mask Scoring R-CNN [14], CenterMask [16], BCNet [15]. The popular examples of one-stage methods are PolarMask [32], YOLOACT [3], and SOLO [28].

3 Experiments

3.1 Datasets

We aim to understand the effects of various state state-of-the-art models on diverse road scene datasets for our experiments. We considered two structured driving datasets; Cityscapes and Berkeley DeepDrive (BDD). In these two datasets, there is low variation in the appearance of objects and also in the background, the road infrastructure is well delineated with proper markings on the road. The same assumptions do not hold for Indian driving conditions. For unstructured driving sequences, we consider the Indian Driving Dataset (IDD) dataset.

Cityscapes *[8]:* is a large scale dataset with urban scenes collected in 50 different cities across Europe. It provides 5000 frames of high-quality pixel-level (fine) annotations and a large set of 20000 weakly (coarse) annotated frames. There are 30 labeled classes, and each image can consist of multiple instances of each class.

Berkeley DeepDrive (BDD) *[34]:* is a diverse and large-scale dataset of visual driving scenes. It consists of over 100K video clips. Each video is about 40 seconds long, 720p, and 30 fps. The videos are recorded using mobile phones, under different weather conditions and are collected from multiple cities in the United States. The dataset is split into training (70K), validation (10K), and testing (20K) sets.

Indian Driving Dataset (IDD) *[27]:* is a dataset of road scenes from unstructured environments in India. It consists of 10004 images, finely annotated with 34 classes collected from 182 drive sequences on Indian roads. A four-level label hierarchy provides varying degrees of complexity. It also has the fallback class to accommodate unknown road objects.

3.2 Setup

For object detection and instance segmentation, we use the popular frameworks Detectron2 [31] and mmdetection [5]. The code is written in PyTorch and executed on a machine with 4 NVIDIA's GeForce GTX 1080 Ti GPUs with CUDA 10.2, CUDNN 7.6.5. Each detector model is trained with a batch of 8 images, learning rate of 0.02, momentum of 0.9, and weight decay factor of 0.0001.

Each instance segmentation model is trained on a base learning rate of 0.01 with other hyper-parameters being the same as the object detection training. For instance segmentation, we train Mask R-CNN [12] model on a ResNet-50 [13] backbone with a base learning rate of 0.01 for 24000 iterations. In Cascaded Mask R-CNN [4] model, we use base learning rate of 0.02 for 27000 iterations. While Mask Scoring R-CNN [14] model has the backbone of ResNeXt-101 [33] with base learning rate of 0.02. We use momentum of 0.9, weight decay of 0.0001, and batch size of 8 for all models.

In the case of semantic segmentation, we use the hyper-parameters as defined in the literature of the respective models. We use two NVIDIA's GeForce GTX 1080 TI GPUs in a Xeon server in order to train the models.

3.3 Performance Metric

We evaluate object detection performance with the widely used mean Average Precision (mAP) [25, 20, 18, 24] metric. We denote AP for bounding box as AP_{box} and mask as AP_{mask}. We also provide class-wise AP_{box} and AP_{mask} for class-wise analysis at a threshold of 0.5. For semantic segmentation evaluation, we use the common metric of mIoU [21, 37, 26]. For instance segmentation evaluation, we use the Average Precision (AP) metric [17, 12, 4]. As in [19], we calculate AP by varying the Intersection over Union (IoU) threshold from 0.5 to 0.95 with a step of 0.05.

4 Results

4.1 Object Detection

Baselines: We choose Faster R-CNN [25] as a two-stage detector, and SSD [20], YOLOv3 [24], and RetinaNet [18] as one-stage detectors for the object detection task.

Table 1. Shows results of object detection: Class-wise Average Precision (AP) on Cityscapes, BDD, and IDD datasets using Faster R-CNN, SSD, YOLOv3, and RetinaNet. The last row indicates mean Average Precision (mAP). **m. cycle:** indicates motorcycle, **t. light:** indicates traffic-light, **t. sign:** indicates traffic-sign, and **veh. flbk:** indicates vehicle-fallback. **FR, Y3, and RN:** indicates Faster R-CNN, YOLOv3, and RetinaNet, respectively. The bold values indicate the category wise best result among all the methods on respective datasets.

	Cityscapes				**BDD**				**IDD**			
Class	**FR**	**SSD**	**Y3**	**RN**	**FR**	**SSD**	**Y3**	**RN**	**FR**	**SSD**	**Y3**	**RN**
person	49.9	30.8	43.5	**51.0**	62.2	45.8	59.1	**65.0**	55.9	41.6	50.2	**57.2**
truck	35.2	32.4	30.9	**38.8**	61.9	59.7	58.7	**63.2**	**68.4**	61.3	57.8	66.8
m. cycle	36.4	32.4	34.8	**41.0**	45.5	34.7	46.0	**46.5**	**70.6**	62.8	63.1	68.6
rider	55.9	37.1	51.3	**56.9**	**48.3**	33.3	47.7	46.8	**59.5**	49.0	54.3	58.1
bus	**63.7**	58.3	58.3	60.9	61.7	60.2	59.4	**62.5**	**74.1**	69.5	67.3	73.5
bicycle	47.4	39.7	44.6	**50.7**	50.0	39.8	46.9	**51.7**	**54.8**	39.3	41.3	52.8
car	67.0	65.0	66.4	**69.8**	79.4	75.9	76.2	**80.6**	**71.0**	65.6	64.5	**71.0**
train	40.9	**47.6**	39.5	45.5	0.0	0.0	**3.3**	0.0	0.0	0.0	0.0	0.0
t. light	-	-	-	-	**64.3**	53.9	57.1	63.1	28.5	13.8	25.5	**29.4**
t. sign	-	-	-	-	**69.8**	64.6	66.5	69.2	**39.5**	27.6	27.3	38.3
caravan	-	-	-	-	-	-	-	-	0.0	0.0	0.0	0.0
auto	-	-	-	-	-	-	-	-	**74.1**	66.9	67.9	73.6
trailer	-	-	-	-	-	-	-	-	0.0	0.0	0.0	0.5
animal	-	-	-	-	-	-	-	-	26.4	20.1	20.2	**28.1**
veh. flbk	-	-	-	-	-	-	-	-	**10.0**	7.9	6.7	**10.0**
mAP	49.6	42.9	46.2	**51.8**	54.3	46.8	52.1	**54.8**	**42.2**	35.0	36.4	41.9

Discussion: Table 1 presents object detection results using Faster R-CNN, SSD, YOLO3, and RetinaNet on Cityscapes, BDD, and IDD datasets. From the table, we observe that RetinaNet performs better than all other detectors on the structured driving datasets: Cityscapes and BDD. While Faster R-CNN obtains the best detection results among all the used models on the unstructured driving dataset: IDD. In the case of IDD, we also observe that all the used methods completely fail to detect objects like *train*, *caravan*, and *trailer* (AP very close to 0). It happens because of fewer amount of annotated images for those categories and unstructured road conditions. For a similar reason, all the methods obtain less than

30% AP scores for object categories such as *traffic-light*, *traffic-sign*, *animal*, and *vehicle-fallback*. We also observe from the table that all models perform better on Cityscapes than IDD and BDD datasets for the object category *train*. This is because of domain shift and ubiquitous presence of the object in Cityscapes than on IDD and BDD. We find similar observation for object categories *traffic-light* and *traffic-sign* on which all the used models perform better on BDD than IDD due to geographic domain shift.

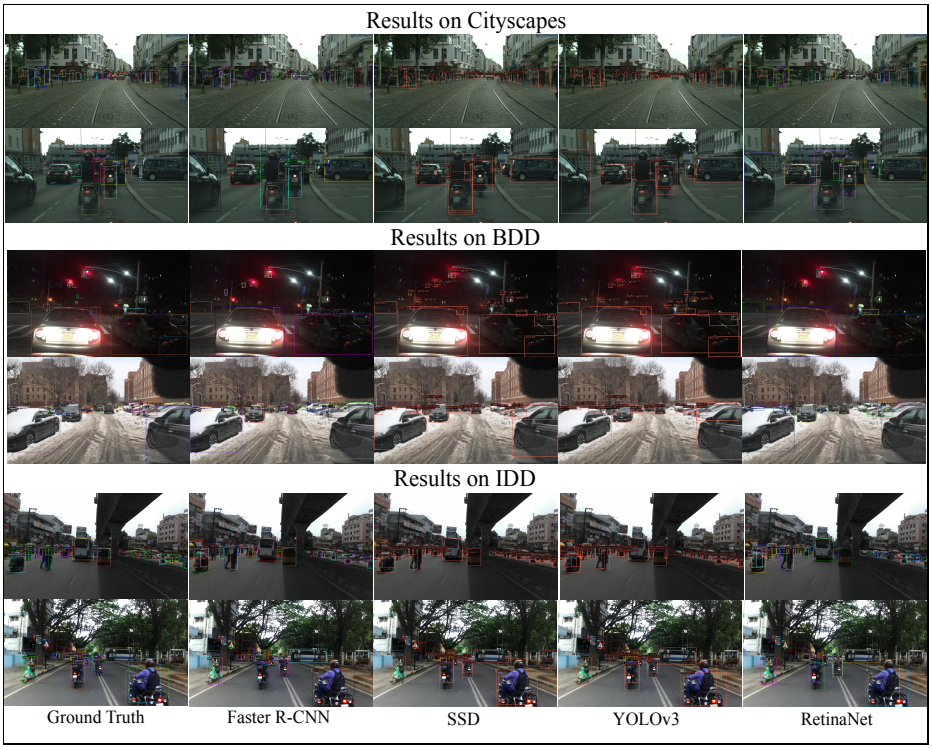

Fig. 2. Presents some qualitative results of object detection on Cityscapes, BDD, and IDD datasets. (Best viewed in color and zoomed).

We present some qualitative results on few selected frames of Cityscapes, BDD, and IDD using the models: Faster R-CNN, SSD, YOLOv3, and RetinaNet in Figure 2. We choose frames under various complex conditions to establish the robustness of the used models. In Cityscapes, one of the selected frames is an empty road with multiple pedestrians walking on the left and right sides of the road, and another image of dense traffic. We notice that Faster R-CNN, YOLOv3, and SSD detect accurate boundaries of all motorcycles and cars. But RetinaNet fails to detect the boundaries of a few cars. In the case of BDD, the selected frames are of moving cars on the road at nighttime and during snowfall. In

both cases, YOLOv3 and SSD detect all cars accurately. While Faster R-CNN and RetinaNet fail to detect cars that are far away from the camera. In nighttime scenarios, we notice some false detection of *traffic-light* and *traffic-sign* caused due to headlights of vehicles. Notice that even in such challenging scenarios, all the models detect all trainable classes. However, Faster R-CNN and RetinaNet fail to detect a few people accurately due to heavy occlusions caused by crowded vehicles and the shadows cast by the trees.

4.2 Semantic Segmentation

Baselines: We choose three popular models, PSPNet, ERFNet, and DRN to benchmark semantic segmentation task on driving sequences.

Discussion: Quantitative score (mIoU) produced by the models are given in Table 2. We train and evaluate a model on various pair-wise combinations of datasets. From the table, we observe that all models achieve the best performance on the Cityscapes dataset. Even the trained model on Cityscapes is often considered a baseline for the segmentation of driving sequences.

It is also interesting to note that the trained model on Cityscapes does not generalize well. Using the model trained on Cityscapes to infer on BDD and IDD (data distribution is different from Cityscapes) resulted in lower performance (almost half of original). From the table, we also infer that Cityscapes is the simplest to learn while IDD is slightly more difficult, and BDD is the most difficult to learn. The trained model on Cityscapes performs poorly on out-of-distribution data points (i.e., IDD and BDD). The trained model on IDD also performs poorly on out-of-distribution data (i.e., BDD and Cityscapes), but it is relatively better than the model trained on Cityscapes. The trained model on BDD performs the best on out-of-distribution data (i.e., IDD and Cityscapes).

Table 2. Shows quantitative results on semantic segmentation: results of three different models: PSPNet, ERFNet, and DRN. The model is trained on one dataset but evaluated on all other three datasets. Values in bold indicates best result among all the methods on respective test dataset.

Training Set	Test Set								
	Cityscapes			BDD			IDD		
	PSP Net	ERF Net	DRN	PSP Net	ERF Net	DRN	PSP Net	ERF Net	DRN
Cityscapes	**76.99**	72.20	71.35	35.06	29.37	38.72	38.46	31.37	40.30
BDD	43.75	33.95	50.77	47.40	37.84	**56.34**	39.70	30.10	46.19
IDD	42.69	28.31	46.43	39.51	28.89	41.91	62.95	59.39	**74.69**

Figure 3 shows visual results of semantic segmentation on Cityscapes, BDD, and IDD datasets using DRN. In the case of Cityscapes, the example images

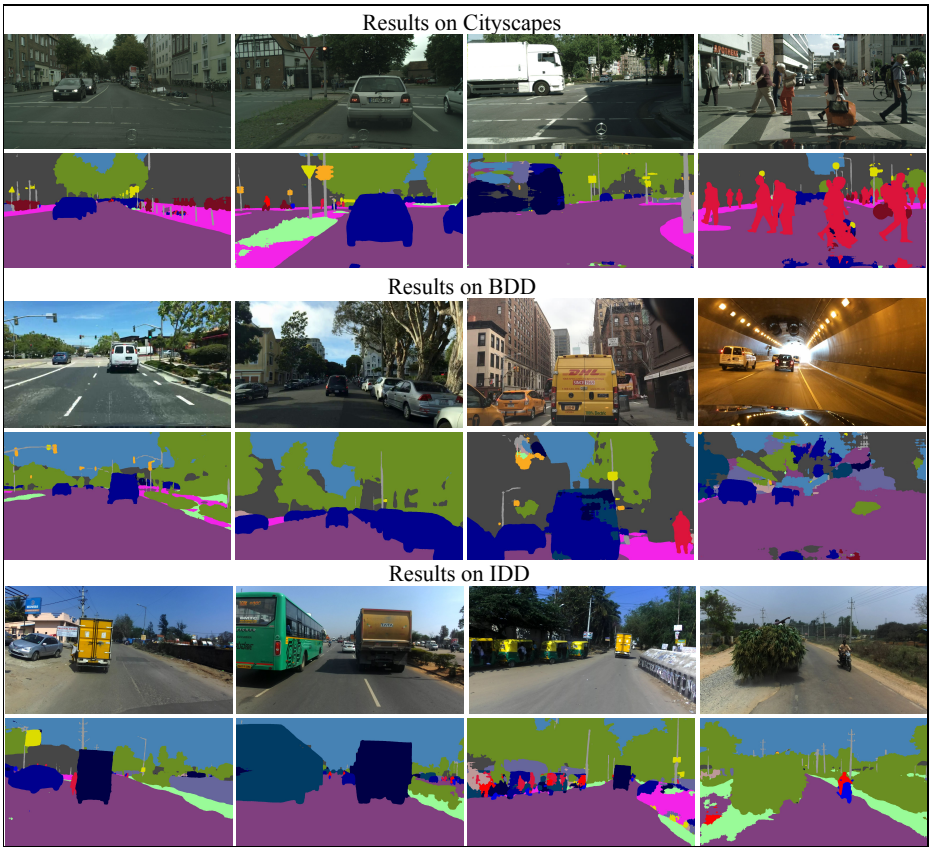

Fig. 3. Presents some qualitative results of semantic segmentation on few selected frames of Cityscapes, BDD, and IDD datasets using DRN-D 38. (Best viewed in color and zoomed).

shown are (i) of a road with few moving cars and has adjacent buildings on both sides, (ii) a road with a single car and dense buildings and trees on both sides, (iii) a big truck is crossing a road with shadow cast by road side's trees and buildings, and (iv) multiple people are crossing a road. DRN produces few false segmentation for all images. While the example images of BDD are of (i) a clean road with few moving cars, (ii) a road with many cars and shadow cast by the roadside adjacent trees, (iii) moving trucks and cars on a road with dense adjacent big buildings, and (iv) car moving under the tunnel. In this case, DRN segments well on the first two images. However, due to dense adjacent buildings and lights in the tunnel, it produces few false segmentation for the third and fourth images. The selected images from IDD contain (i) a clean road with two cars, (ii) road with a bus and a truck, overtaking each other, (iii) a road with dense autos and dense trees on the road's side, and (iv) road with one moving

motorcycle. DRN performs reasonably well for all images except the third image where the performance drops due to dense vehicles and adjacent roadside trees.

4.3 Instance Segmentation

Baselines: We use three popular existing models — Mask R-CNN (MR), Cascade R-CNN (CR), and Mask Scoring R-CNN (MSR) to benchmark instance segmentation tasks on driving datasets.

Table 3. Shows results on instance segmentation: Class-wise AP_{box} and AP_{mask} scores on Cityscapes, BDD, and IDD datasets. **MR, CM, MSR:** indicates Mask R-CNN, Cascaded Mask R-CNN, and Mask Scoring R-CNN, respectively. Values in bold indicates the best results among all the methods on respective datasets.

		Cityscapes			**BDD**			**IDD**		
Class	**Metric**	**MR**	**CM**	**MSR**	**MR**	**CM**	**MSR**	**MR**	**CM**	**MSR**
person	AP_{box}	**41.6**	32.3	33.8	31.0	**37.3**	32.7	**35.6**	31.2	29.1
	AP_{mask}	**34.0**	23.9	25.7	25.3	**32.7**	30.2	**31.5**	27.1	25.2
truck	AP_{box}	**35.2**	23.7	25.7	28.8	**34.0**	28.2	**54.1**	52.7	49.5
	AP_{mask}	**35.7**	23.6	26.8	27.9	**33.4**	27.9	**53.1**	50.3	49.6
motorcycle	AP_{box}	**29.5**	18.4	23.7	25.3	28.7	**28.9**	**39.9**	38.5	35.1
	AP_{mask}	**22.5**	12.6	15.5	15.5	16.8	**17.8**	**32.2**	30.6	28.2
rider	AP_{box}	**43.7**	36.9	37.6	21.0	20.5	**21.6**	**39.5**	37.4	34.3
	AP_{mask}	**29.2**	21.0	21.8	08.8	11.7	**11.8**	**29.4**	27.3	24.4
bus	AP_{box}	**60.1**	52.5	42.5	30.8	**35.5**	27.7	**49.5**	48.1	43.4
	AP_{mask}	**58.8**	49.5	42.0	30.0	**35.4**	28.9	**47.9**	45.3	43.8
bicycle	AP_{box}	**34.7**	25.9	30.4	11.5	**13.5**	10.7	**24.3**	20.5	20.4
	AP_{mask}	**22.9**	15.7	15.8	05.6	**08.3**	07.9	**14.3**	12.1	11.7
train	AP_{box}	**28.8**	14.7	9.5	0.0	0.0	0.0	-	-	-
	AP_{mask}	**42.1**	25.2	13.2	0.0	0.0	0.0	-	-	-
car	AP_{box}	**58.4**	50.8	52.6	48.0	**52.4**	47.6	**54.1**	51.9	48.6
	AP_{mask}	**52.5**	44.2	45.1	44.1	**48.6**	45.4	**50.2**	47.1	45.4
autorickshaw	AP_{box}	-	-	-	-	-	-	**55.5**	53.9	50.0
	AP_{mask}	-	-	-	-	-	-	**52.4**	48.9	47.1
vehicle-fallback	AP_{box}	-	-	-	-	-	-	**04.5**	03.8	02.8
	AP_{mask}	-	-	-	-	-	-	**03.9**	03.2	02.5
Average	AP_{box}	**41.5**	31.9	31.9	24.5	**27.7**	24.6	**39.7**	37.5	34.8
	AP_{mask}	**37.2**	26.9	25.7	19.7	**23.3**	21.2	**34.9**	32.4	30.9

Discussion: Table 3 shows the class-wise AP_{box} and AP_{mask} scores on Cityscapes, BDD, and IDD datasets. We notice that for all object categories, Mask R-CNN obtains the best AP_box and AP_mask scores among all methods for Cityscapes and IDD. While Cascade Mask R-CNN obtains the best AP_{box} and AP_{mask} scores for majority of object classes (except *motorcycle* and *rider*) in case of BDD. For

Cityscapes, Mask Scoring R-CNN produces the worse AP_{box} and AP_{mask} scores which are 9.5% and 11.4% lower than that of Mask R-CNN.

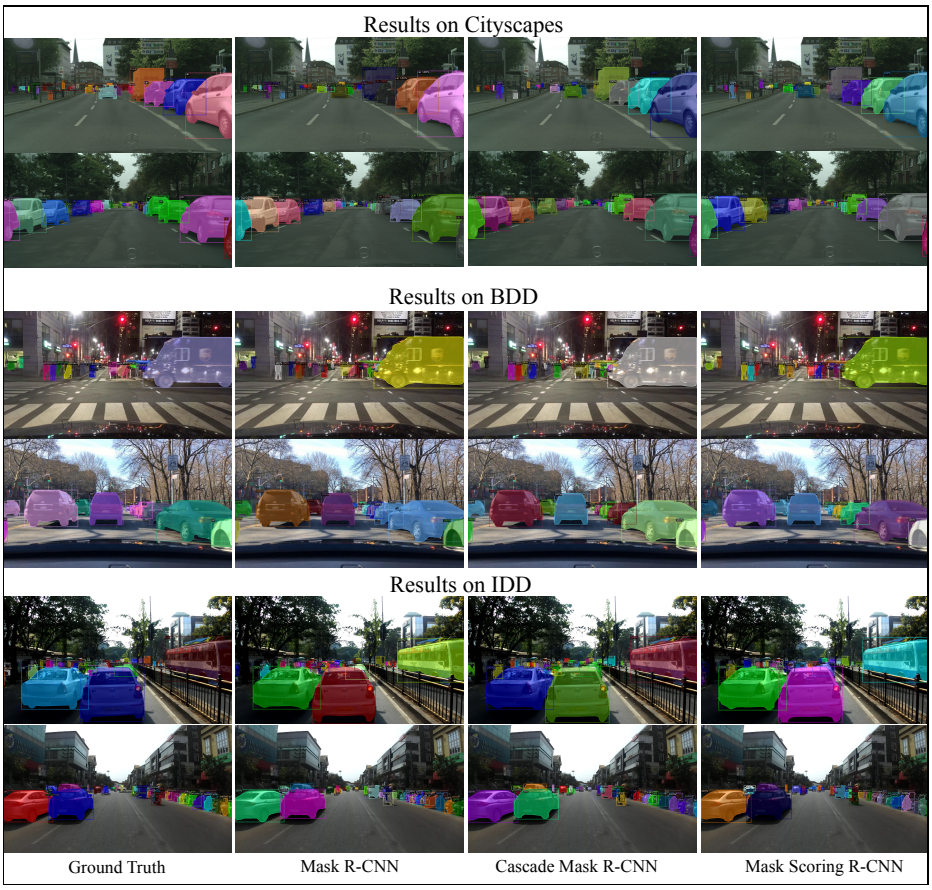

Fig. 4. Shows some qualitative results on instance segmentation showing both mask and bounding boxes of a few selected frames from Cityscapes, BDD, and IDD datasets. (Best viewed in color and zoomed).

For *motorcycle* and *train* object categories of Cityscapes, the performances of the two techniques — Cascade Mask R-CNN and Mask Scoring R-CNN drops more than 10% compared to Mask R-CNN. In BDD, we notice that for the object category car, all methods obtain AP_{box} and AP_{mask} scores more than 44%. We also notice that Cascade Mask R-CNN obtains the best average AP_{box} and AP_{mask} and Mask R-CNN obtains the worse results among the used methods. In case of IDD, we notice that the AP_{box} score is higher (more than 39%) for commonly found road objects such as *autorickshaw*, *truck*, *bus*, and *car*. While

the AP_{mask} score for *person* and *rider* classes is very alike due to similar looking visual features. Among the object categories, all methods achieve the lowest AP_{box} and AP_{mask} for *vehicle-fallback*. MR obtains the best AP_{box} and AP_{mask} scores compared to other techniques. While MSR produces the worse average AP_{box} and AP_{mask} scores which are 4.8% and 4%, respectively less than MR.

From the table, we also observe that the performances of all methods are worse on BDD as compared to Cityscapes and IDD. This is because sequences of BDD are captured under several complex conditions. The quantitative results highlight that BDD is more complex than IDD and Cityscapes for instance segmentation. While IDD is more complex than Cityscapes for the same task.

Figure 4 shows the qualitative results of a few randomly selected frames. Images of the first and second rows are from Cityscapes. Both images show multiple cars moving on the left and right sides of the road. From the figure, we notice that Mask R-CNN produces better results on overlapping cars. However, both Cascade Mask R-CNN and Mask Scoring R-CNN fail to segment instances of a car far away from the camera. The image in the third row shows a crowded traffic junction and multiple pedestrians are crossing the road in BDD. We notice that Cascade Mask R-CNN can accurately segment instances of small pedestrians than Mask R-CNN and Mask Scoring R-CNN. Images of the fourth row present multiple moving cars on a road, with shadows cast by roadside trees. In this case, Mask Scoring R-CNN obtains the best results. Images of fifth and sixth rows are taken from IDD and include multiple overlapping vehicles of varying scales on a road with dense buildings on both sides. In both the cases, Mask R-CNN produces better results than Cascade Mask R-CNN and Mask scoring R-CNN.

5 Summary

In this work, we used various state-of-the-art models for object detection, semantic segmentation, and instance segmentation tasks and evaluate their characteristics on structured and unstructured driving datasets: Cityscapes, BDD, and IDD. To our knowledge, this work is the first comprehensive report on analyses of models for tasks with driving datasets. All the methods performed significantly better on object category *train* in Cityscapes than on BDD and IDD in the object detection task. Due to the unstructured nature, object detection tasks on IDD performed lower compared to Cityscapes and BDD. Cityscapes is the easiest dataset for object detection tasks among the three datasets being used. In semantic segmentation, we notice that all models perform better on Cityscapes than on BDD and IDD. We also notice that the DRN model performs consistently well across all the driving datasets compared to other models. In instance segmentation, we observe that Mask R-CNN performs better than all other models on Cityscapes and IDD, while Cascade Mask R-CNN performs better for the majority of the object categories of BDD. Looking at the complexity of the dataset for different tasks, we notice that for instance segmentation and semantic segmentation tasks, BDD is a more complex dataset than Cityscapes and IDD.

A further study on identifying and addressing the problems inherent in road scene datasets can help in a better generalization. An empirical study on domain adaptation and domain generalization can be performed to further understand the behavior of the models in different geographic and environmental settings.

Acknowledgements: This work was partly funded by IHub-Data at IIIT-Hyderabad.

References

1. Araki, R., Onishi, T., Hirakawa, T., Yamashita, T., Fujiyoshi, H.: MT-DSSD: Deconvolutional single shot detector using multi task learning for object detection, segmentation, and grasping detection. In: ICRA (2020)
2. Arnab, A., Jayasumana, S., Zheng, S., Torr, P.H.: Higher order Conditional Random Fields in deep neural networks. In: ECCV (2016)
3. Bolya, D., Zhou, C., Xiao, F., Lee, Y.: YOLACT: Real-time instance segmentation. In: ICCV (2019)
4. Cai, Z., Vasconcelos, N.: Cascade R-CNN: High quality object detection and instance segmentation. IEEE Trans. on PAMI (2019)
5. Chen, K., Wang, J., Pang, J., Cao, Y., Xiong, Y., Li, X., Sun, S., Feng, W., Liu, Z., Xu, J., Zhang, Z., Cheng, D., Zhu, C., Cheng, T., Zhao, Q., Li, B., Lu, X., Zhu, R., Wu, Y., Dai, J., Wang, J., Shi, J., Ouyang, W., Loy, C.C., Lin, D.: MMDetection: Open MMLab detection toolbox and benchmark. arXiv (2019)
6. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. IEEE Trans. on PAMI (2018)
7. Chen, L.C., Yang, Y., Wang, J., Xu, W., Yuille, A.L.: Attention to scale: Scale-aware semantic image segmentation. In: CVPR (2016)
8. Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., Schiele, B.: The Cityscapes dataset for semantic urban scene understanding. In: CVPR (2016)
9. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. IJCV (2010)
10. Girshick, R.: Fast R-CNN. In: ICCV (2015)
11. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: CVPR (2014)
12. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask R-CNN. In: CVPR (2017)
13. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
14. Huang, Z., Huang, L., Gong, Y., Huang, C., Wang, X.: Mask Scoring R-CNN. In: CVPR (2019)
15. Ke, L., Tai, Y.W., Tang, C.K.: Deep occlusion-aware instance segmentation with overlapping bilayers. In: CVPR (2021)
16. Lee, Y., Park, J.: Centermask: Real-time anchor-free instance segmentation. In: CVPR (2020)
17. Liang, X., Lin, L., Wei, Y., Shen, X., Yang, J., Yan, S.: Proposal-free network for instance-level object segmentation. IEEE trans. on PAMI (2017)
18. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: ICCV (2017)

19. 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: ECCV (2014)
20. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: SSD: Single shot multibox detector. In: ECCV (2015)
21. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: CVPR (2015)
22. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: CVPR (2016)
23. Redmon, J., Farhadi, A.: YOLO9000: Better, faster, stronger. In: CVPR (2017)
24. Redmon, J., Farhadi, A.: YOLOv3: An incremental improvement. arXiv (2018)
25. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: Towards real-time object detection with region proposal networks. In: NeurIPS (2015)
26. Romera, E., Alvarez, J.M., Bergasa, L.M., Arroyo, R.: ERFNet: Efficient residual factorized convnet for real-time semantic segmentation. IEEE Trans. on Intelligent Transportation Systems (2018)
27. Varma, G., Subramanian, A., Namboodiri, A., Chandraker, M., Jawahar, C.: IDD: A dataset for exploring problems of autonomous navigation in unconstrained environments. In: WACV (2019)
28. Wang, X., Kong, T., Shen, C., Jiang, Y., Li, L.: SOLO: Segmenting objects by locations. In: ECCV (2020)
29. Wang, Y., Zhou, Q., Xiong, J., Wu, X., Jin, X.: ESNet: An efficient symmetric network for real-time semantic segmentation. In: PRCV (2019)
30. Wu, T., Tang, S., Zhang, R., Cao, J., Zhang, Y.: CGNet: A light-weight context guided network for semantic segmentation. IEEE Trans. on Image Processing (2020)
31. Wu, Y., Kirillov, A., Massa, F., Lo, W.Y., Girshick, R.: Detectron2. https://github.com/facebookresearch/detectron2 (2019)
32. Xie, E., Sun, P., Song, X., Wang, W., Liu, X., Liang, D., Shen, C., Luo, P.: Polarmask: Single shot instance segmentation with polar representation. In: CVPR (2020)
33. Xie, S., Girshick, R., Dollár, P., Tu, Z., He, K.: Aggregated residual transformations for deep neural networks. In: CVPR (2017)
34. Yu, F., Chen, H., Wang, X., Xian, W., Chen, Y., Liu, F., Madhavan, V., Darrell, T.: BDD100k: A diverse driving dataset for heterogeneous multitask learning. In: CVPR (2020)
35. Yu, F., Koltun, V., Funkhouser, T.A.: Dilated residual networks. arXiv (2017)
36. Zhang, S., Wen, L., Bian, X., Lei, Z., Li, S.Z.: Single-shot refinement neural network for object detection. In: CVPR (2018)
37. Zhao, H., Shi, J., Qi, X., Wang, X., Jia, J.: Pyramid scene parsing network. In: CVPR (2017)
38. Zhao, Q., Sheng, T., Wang, Y., Tang, Z., Chen, Y., Cai, L., Ling, H.: M2Det: A single-shot object detector based on multi-level feature pyramid network. In: AAAI (2019)