Traffic monitoring system based on vehicle detection and tracking algorithms in smart city

Hoang-Thong Vo[1,*], Ngan-Linh Nguyen[1,*], Dinh-Tu Vo[1,*], Thien-Long Nguyen[1,*], and
Trong-Hop Do[1,†]

[1]University of Infomation Technology, Ho Chi Minh City, Vietnam
[*] *{18521462, 18520989, 18521589, 18521046}@gm.uit.edu.vn*
[†] *{hopdt}@uit.edu.vn*

Abstract. Transport systems in smart cities are one of the research areas that have received a lot of attention in recent years. In this study, we propose a method to build traffic monitoring system using object detection and tracking algorithms. Object detection models YOLOv4, YOLOv5 and DeepSort were installed and evaluated on the UIT-DET dataset. The model's performance is relative on the UIT-DET dataset with real-world contexts and complexity.

Keywords: Multiple-Object Tracking · DeepSort · CenterTrack · Traffic monitoring system, Multiple-Object Detection.

1 Introduction

Traffic congestion has long been a problematic hassle in building any smart city all around the world. Worn-out infrastructure, increasing urban populations, inefficient and uncoordinated traffic signal timing, and a lack of real-time data as well as the help from advanced machine learning are among the most astounding contributing factors to traffic congestion.

The effects of traffic congestion strikes hard on both financial aspects and life. Traffic congestion cost U.S. commuters $305 billion in 2017 due to wasted fuel, lost time and the increased cost of transporting goods through congested areas – as estimated by Traffic Data and Analytics company INRIX [1]. On account of the physical and financial limitations around building additional roads, cities must now use new strategies and technologies to improve traffic conditions. And the ultimate solution comes from nothing but the advanced traffic management technologies such as adaptive traffic control and traffic analytics which can improve safety and significantly decrease traffic congestion levels and greenhouse gas (GHG) emissions.

Our work aims to undertake on the very first step of traffic analytics which is counting vehicles on every traffic route throughout the city. We use CenterTrack [2] and DeepSort [3] as real-time tracking methods to exercise on our own dataset - the UIT-DET dataset. Our dataset used to be trained on detection models to simply detect and classify vehicle classes, though to exercise vehicle counting, becomes kind of meaningless. Below is what we have got to say on why tracking is the way to go instead of detection.

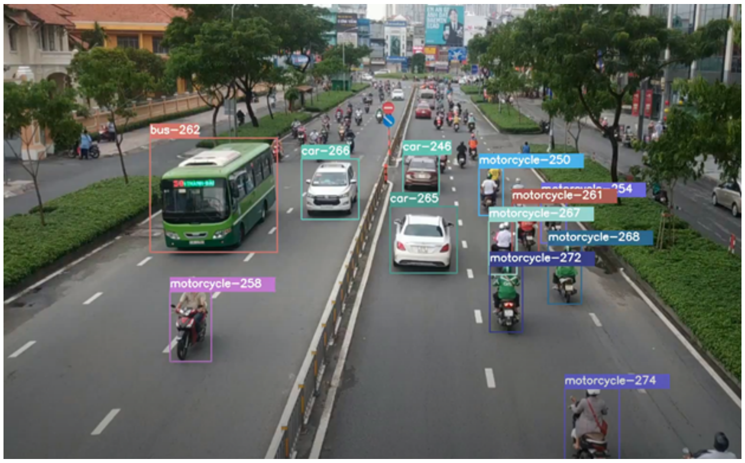

Fig. 1: Each vehicle has its own class and ID tracked

Vehicle detection can perform counting vehicles on a single image, however, when considering our task with real-time footage it seems to lose consistency in terms of both overall count and per-class count, that said apparently vehicle detection can only be used on counting vehicles at specific time points. That is when tracking vehicles comes in handy, a good tracking model can determine the exact number of vehicles from each class that exists on any traffic route during a set time interval δt. Given a line mark on a specific route, and a time point t_1, each vehicle that passes the line has its own ID and class which helps us calculate the exact number of vehicles that pass the route upon reaching the end time point t_2 ($\delta t = t_2 - t_1$).

2 Related work

In the research to find out effective traffic management methods, some authors have proposed more optimal management of vehicle traffic and traffic light management by the modern technology platform. for example, In Djahel S et al. proposed a solution [4] to improve performance of Intelligent Transport System (ITS) for applications that require proper dissemination of event driven warning messages. While Yan G et al. promoted the vision of Vehicular Clouds (VCs) and various security challenges in vehicular clouds. [5]. In addition Wen W. proposed a prototype for automatic traffic light control expert system [6]. Its simulation model consists of various sub models. The model simulates the arriving and leaving number of vehicles on roads by using inter-arrival and inter-departure time. Yu R et al. proposed a model to integrate cloud computing and vehicular networks to share computational resources [7]. And we can't be ignored that the proposed architecture consists of central, vehicular and roadside cloud. Li J et al. presented rule based iterative Artificial Transportation System (ATS) design process [8]. In ATS traffic simulations are done in synthetic way to deal with traffic issues from complex system point of view.

On the other hand, some people think of doing research on problems related to traffic jams and traffic accidents like Dong CF et al. with a new strategy named Weighted Congestion Coefficient Feedback Strategy (WCCFS) [9]. Through this technique any dynamic information can be produced and shown to guide the users on road. Moreover, Lee JK et al. proposed a service based Intelligent Transportation System Framework (s-ITSF) [10] to provide efficient accident management while Sumra IA et al. proposed a Vehicular SMS System (VSS) [11] in order to problems such as traffic jams and road accidents.

Last but not least, there are people who think about overcoming the social and environmental problems caused by traffic. Alsabaan et al. focused on creating an Economical and Environment Friendly Geo cast (EEFG) protocols [12] to minimize fuel consumptions and emissions. They proposed the method to integrate vehicular networks with fuel models.

3 Methodology

3.1 YOLO

You-Only-Look-Once (YOLO), a Convolution Neral network dedicated to solving real-time object detection problems, is designed by Joseph Redmon et al since 2015[13]. The idea is bringing on the process of regression model for object prediction. YOLO uses a single neural network architecture to predict the bounding box and determine the classifier output from the image.

The most outstanding advantage of YOLO is that it only needs to use the entire image information once and predict the entire object box containing the objects, the algorithm model is built in an end-to-end style so that the training is completely by gradient descent.

3.1.1 YOLOv3

Continuing to build on previous achievements, Joseph Redmon and Ali Farhadi from the University of washington published a scientific report on an improved YOLO model in April 2018 called YOLO version 3 [14]

Unlike version 2, YOLOv3 uses the darknet53 model (YOLOv2 uses darknet19). This means that the author has added 53 convolutional layers to the CNN model (raising the total number of convolutional layers to 106). In addition, in image processing capabilities, YOLOv3 processes by detecting 3 times on 1 frame with 3 different sizes in convolutional layers (82, 94 and 106) in the direction of increasing image size.

In addition, the author also added a new concept called Anchor box. they are bounding boxes but are pre-made. the training process combined with the Kmean cluster algorithm will generate these boxes.

However, in the improvement process, there are still some efforts that not only do not bring the desired results, but also cause damage:

- Using linear activation to predict the position information of the anchor box leads to a decrease in the stability of the training model.
- Using Focal loss increased in a 2 point reduction in mAP.
- Using Fast-RCNN's dual IOU strategy to predict ground truth has no effect.

3.1.2 YOLOv4

As a new version of YOLO with a series of speed improvements over its predecessors and installed from another version of Darknet, YOLOv4 was developed by Alexey Bochkovskiy based on previous versions of YOLO by Joseph Redmon[15].

The architecture of the YOLOv4 model helps programmers to approach object detection problems without requiring strong computational resources. In addition, it can train an object detection network with very high accuracy with just a 1080ti or 2080ti GPU.

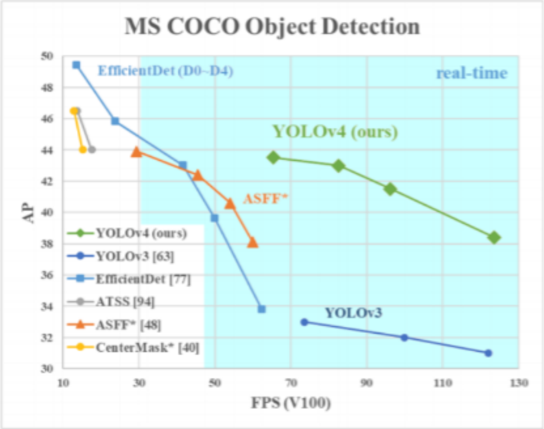

Fig. 2: The results of comparing YOLOv4 with current SOTA

According to the results tested in Figure 2, YOLOv4 runs twice as fast as EfficientDet, increasing AP and FPS compared to YOLOv3 by 10 % and 12 %, respectively. More specifically, YOLOv4 reached 43.5 % AP on MS COCO dataset at 65 FPS (performed on Tesla V100 GPU).

To achieve the above results, the author used a series of techniques improved over previous versions, including: Weighted-Residual-Connections (WRC), Cross-Stage-Partial- connections (CSP), Cross mini-Batch Normalization (CmBN), Self-adversarial-training (SAT) and Mish-activation, Mosaic data augmentation, DropBlock regularization, and CIoU loss.

3.2 DeepSORT - The improved version of SORT

Simple Online Real Time Object Tracking (SORT) is a type of Detection Based Tracking developed by Alex Zongyuan Ge et al since Feb 2, 2016 [16]. It is a tracking algorithm that supports support for object decimation algorithms, using pre-trained weights. The idea of the algorithm is to detect the detached object as a separate problem then find a way to associate the bound boxes captured in each frame and the ID size for each object.

Specifically, the prediction step will play the role of predicting the new position of the object based on the previous frame. The linking step then associates the detected locations with the location prediction assigned to the corresponding ID. To implement that idea, the authors uses Hungary algorithm and kalman filter.

To improve the limitations of SORT, Nicolai Wojke et al developed an improved SORT model called DeepSORT [3]. This is a new version of SORT to solve the problem of high number of ID switches. The idea of algorithm is based on using deep learning to extract features of objects to increase accuracy in data association. In addition, a linking strategy was also built called Matching Cascade to help link objects after disappearing for a while more effectively.

These DeepSORT's innovations help to limit the disadvantages of SORT model. However, in order to trade off higher performance, DeepSORT has a slower processing speed but it is still fast enough to be able to support object detection algorithms in real-time tracking.

3.3 CenterTrack - Tracking Objects and Points

3.3.1 CenterNet - Objects and Points

Our approach is the Object Tracking problem. The proposed method is called CenterTrack, to better understand the algorithmic starting point of the method, we need to learn an Object Detection: CenterNet[17] network model. The model was released in 2019, the model has an extremely simple design, but achieves a good balance between speed and accuracy, becoming a state of the art and quickly receiving recognition from community.

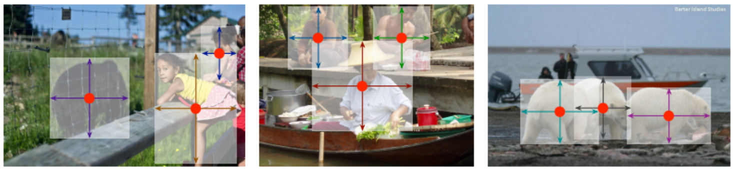

Fig. 3: Visualize the CenterNet model - Object As Point

Another special feature of CenterNet introduced can solve for 3 different tasks: Object Detection, Human Pose Estimation and 3D Object Detection, but in this paper we only mention Object Detection, this task is visualized in figure 3.

The model will learn how to generate heatmaps and then generate heatmap ground truths to help the model compare and optimize during backpropagation. Specifically in the figure 4, with an input image $I \in R^{WxHx3}$, the output will be a heatmap $Y \in [0,1]^{\frac{W}{R}x\frac{H}{R}xC}$ with R being stride (defined before, in paper, the author uses R = 4), and C is the number of classes (in our problem is 5). In this CenterNet, we use stride R=4, so the heatmap size will be 4 times smaller than the input image. Since we are doing object detection problem with class number of 5, input image 512x512x3 (3 is number of color channels in RGB color system). Thus, the heatmap will be 128x128x5 in size.

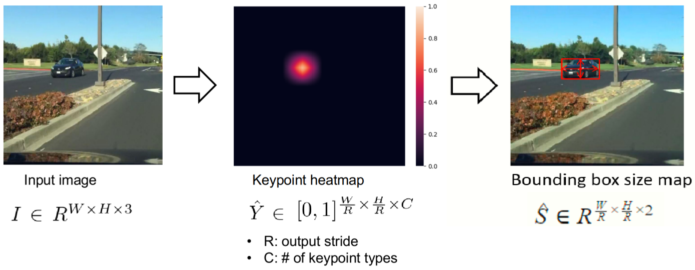
1.0
0.8
0.6
0.4
0.2
0.0
Input image
$I \in R^{W\times H\times 3}$
Keypoint heatmap
$\hat{Y} \in [0,1]^{\frac{W}{R}\times\frac{H}{R}\times C}$
R: output stride
C: # of keypoint types
Bounding box size map
$\hat{S} \in R^{\frac{W}{R}\times\frac{H}{R}\times 2}$

Fig. 4: Visualize framework of the CenterNet model

In the last figure in figure 4, parallel to estimating the position of the center (keypiont of the detected object), it is also estimating the size of the detected object, namely the width (w) and height (h). CenterNet also uses a head - dimension head [17] to estimate the width and height of the object. Output of w,h head is 1 tensor $S \in R^{\frac{W}{R}x\frac{H}{R}x2}$. From there we get the bounding box along with the detected object's keypoint, in preparation for the Object Tracking - CenterTrack method in the next section 3.3.2 below.

3.3.2 CenterTrack - Tracking method based on CenterNet

Objective of method: We approach the CenterTrack method similar to the approach that the author introduced in their paper[2], with a point-based framework for simultaneous detection and tracking with the goal of simplified Multi-object Tracking (MOT).

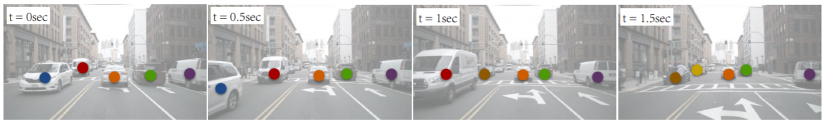

Fig. 5: Visualize CenterTrack method based on CenterNet algorithm

We observe figure 5. Based on CenterNet, CenterTrack represents each object with a single point at the center of its bounding box. However, CenterNet cannot find objects that are not visible in the current frame. To increase temporal coherence, CenterTrack is provided with two consecutive frames as input. This enables CenterTrack to estimate the change in scene and recover occluded objects in the current frame with visual evidence in the previous frame. A class-agnostic heatmap of prior tracked object centers represented as points is also provided. This algorithm is more specific to the framework shown in figure 6 below.

Framework of method:

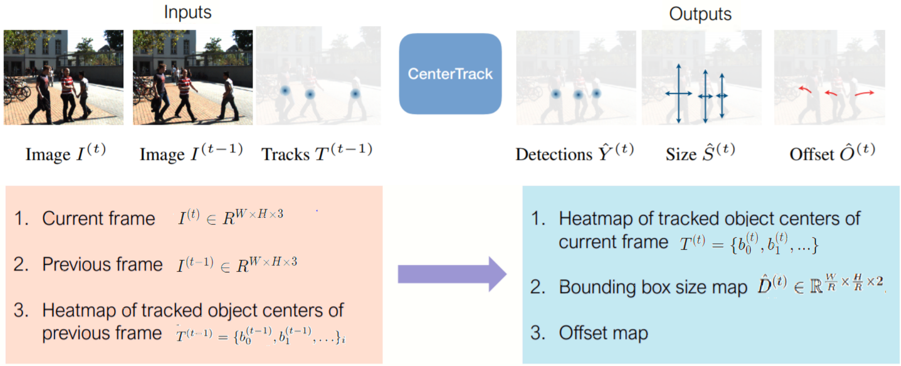
Inputs
Outputs
CenterTrack
Image $I^{(t)}$
Image $I^{(t-1)}$
Tracks $T^{(t-1)}$
Detections $\hat{Y}^{(t)}$
Size $\hat{S}^{(t)}$
Offset $\hat{O}^{(t)}$
1. Current frame $I^{(t)} \in R^{W \times H \times 3}$
2. Previous frame $I^{(t-1)} \in R^{W \times H \times 3}$
3. Heatmap of tracked object centers of previous frame $T^{(t-1)} = \{b_0^{(t-1)}, b_1^{(t-1)}, \ldots\}_i$
1. Heatmap of tracked object centers of current frame $T^{(t)} = \{b_0^{(t)}, b_1^{(t)}, ...\}$
2. Bounding box size map $\hat{D}^{(t)} \in \mathbb{R}^{\frac{W}{R} \times \frac{H}{R} \times 2}$
3. Offset map

Fig. 6: Framework of CenterTrack method

At the output, the offset between the current object center and its center in the previous frame is learned as an attribute of that center point. Object association is based on the distance between the object's center point in the previous frame and the position at the predicted offset from the object's current center point. Using greedy matching, each current detection is associated with the closest unmatched prior detection based on the above distance. A new object is spawned when there is no unmatched prior detection within a certain radius.

4 Experiments

4.1 Dataset

A UIT-DET is a dataset of traffic scenes in Vietnam, collected from many locations in big cities. Data sets with diverse contexts and complexity. Same as object detection benchmark datasets like COCO, PASCAL VOC, KITTI, MOT16, and UA-DET [18–22]. Our dataset is built for vehicle detection and tracking problem for smart traffic systems in a smart city.

The UIT-DET dataset is divided into 3 sets, the training set (UIT-DET-train), the calibration set (UIT-DET-validation), and the test set (UIT-DET-test) are divided proportionally. are 0.7, 0.2, and 0.1 on the UIT-DET dataset, respectively. In Figure 7, the distribution of the classes of the data is depicted.

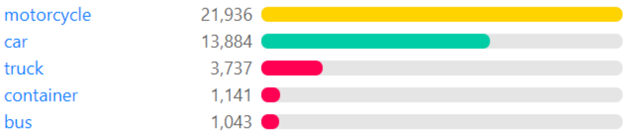

Fig. 7: Distribution of model numbers in classes of motorcycle, car, bus, truck, container

4.2 Evaluation metric

Based on the [23–25] studies on the evaluation metrics for the object detection and tracking problem, we use the metrics mAP, IoU, MOTA, MOTP, IDF1, MT, ML, HOTA to evaluate the performance of state-of-the-art object detection and tracking algorithms.

4.3 Result

For the object detection problem, we install and implement two state-of-the-art models, YOLOv4 and YOLOv5. The results from Table 1 show that the YOLOv5 algorithm gives a high performance on the evaluation dataset with an accuracy of over 80%. In practice, the model does not perform well and there are many data points the model predicts incorrectly.

For the object tracking problem, we use DeepSort algorithm combined with YOLOv5 detector to evaluate on the UIT-DET dataset with different measures

Table 1: *Method of object detection based on mAP . evaluation metrics*

Model	Recall	Precision	mAP@0.5	mAP@0.75	mAP@0.5:0.95
YOLOv4	0.78	0.74	0.833	-	-
YOLOv5	0.996	0.721	0.981	-	0.836

as shown in Table 2. In this study, we can only install YOLOv5 combined with DeepSort for evaluation, because to implement the above mentioned algorithms is extremely difficult. In Figure 8, the model results for detecting and tracing vehicles.

Table 2: *Object tracking method based on evaluation metrics MOTA, MOTP, IDF1, MT, ML, HOTA*

Model	MOTA	MOTP	IDF1	MT	ML	HOTA
YOLOv5_deepsort	59.683	66.419	77.461	22	4	48.065

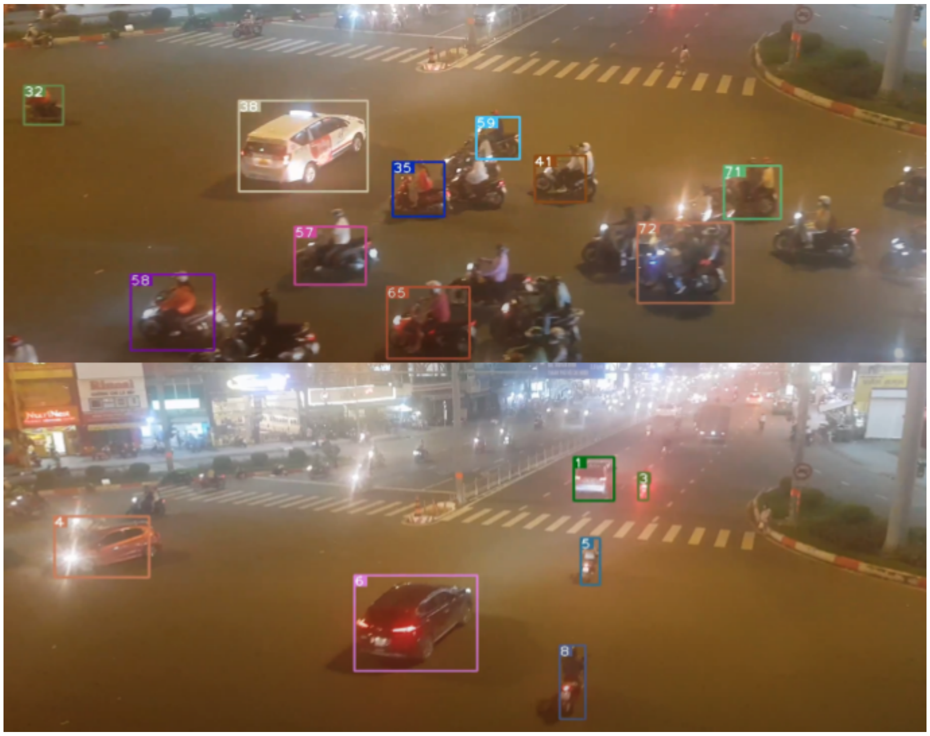

Fig. 8: Vehicle tracking on the UIT-DET dataset

4.4 Error analysis

In Figure 9, car-1 and car-5 are the same car, but only needing two different shapes, the model misrecognizes the results. Although the vehicle detection model worked well, the DeepSort model failed to associate previous frames to identify the vehicle. The tracking problem is a challenging problem, current studies still achieve certain achievements on the benchmark dataset, but it still takes time to achieve good results.

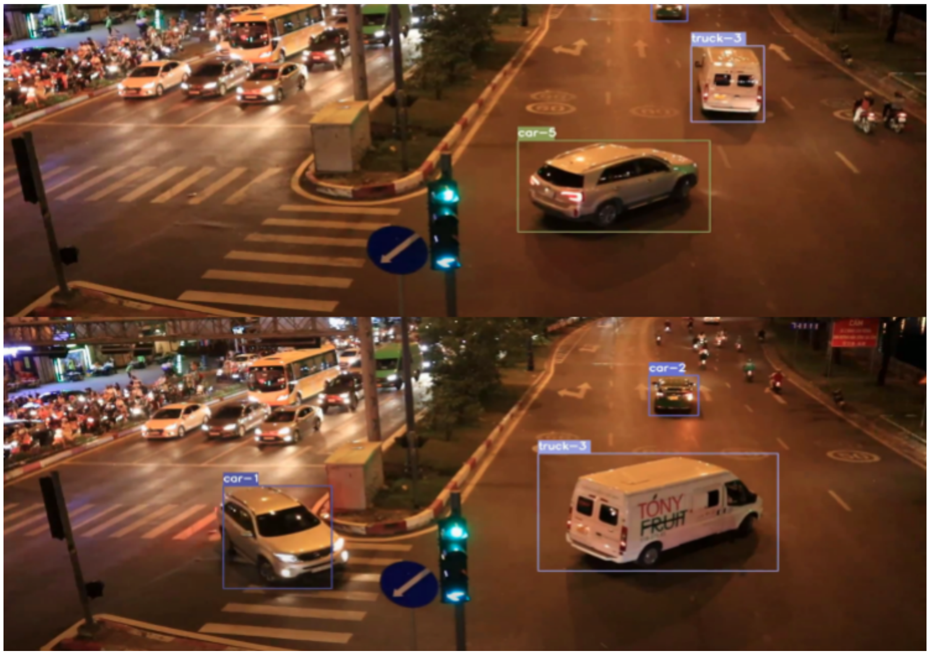

Fig. 9: Error analysis on the UIT-DET dataset

5 Conclusion

We have successfully built a vehicle tracking system in a smart city based on the UIT-DET dataset using current well-known Object Tracking methods such as DeepSort, CenterTrack. Along with this work, we also got demo results on unlabelled street traffic videos, and more importantly got the results of our Tracking method evaluation, namely the results of DeepSort method based on YOLOv5 algorithm.

The results of the Tracking method with the MOTP measurement above 65%, along with the MOTA measurement reaching nearly 60%, and IDF1 reaching 77%. This is a pretty good result compared to our predictions for our traffic tracing. However, when compared to other Baseline of many Object Tracking projects, it is still not high and standard. This problem we still need to overcome, the main reason is that the data set is not optimal and complete, along with it is that the labeling work is still limited. The difficulties we will solve in future works need to reuse this our dataset.

The difficulty with using CenterTrack based on CenterNet method is that we have not yet got the evaluation results in Multiple-Object Tracking work, even though we have demoed the method using the trained weights of the method on your custom dataset. We put this difficulty into the work that needs to be solved in the future. At the same time, we apologize for this inconvenience. However, we have learned and researched a new Object Tracking method mentioned in detail in this paper, different from DeepSort based on YOLO algorithm.

References

[1] INRIX, *Los angeles tops inrix global congestion ranking*, (2018) `https://inrix.com/press-releases/scorecard-2017/` (visited on 07/08/2021).

[2] X. Zhou, V. Koltun, and P. Krähenbühl, "Tracking objects as points", (2020).

[3] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric", (2017).

[4] S. Djahel, R. Doolan, G.-M. Muntean, and J. Murphy, "A communications-oriented perspective on traffic management systems for smart cities: challenges and innovative approaches", IEEE Communications Surveys Tutorials **17**, 125–151 (2015).

[5] G. Yan, D. Wen, S. Olariu, and M. C. Weigle, "Security challenges in vehicular cloud computing", IEEE Transactions on Intelligent Transportation Systems **14**, 284–294 (2013).

[6] W. Wen, "A dynamic and automatic traffic light control system for solving the road congestion problem", Expert Systems with Applications **34**, 2370–2381 (2008).

[7] Y. Zhang, S. Gjessing, W. Xia, and K. Yang, "Toward cloud-based vehicular networks with efficient resource management", Network, IEEE **27**, `10.1109/MNET.2013.6616115` (2013).

[8] J. Li, S. Tang, X. Wang, and W. Duan, "Growing artificial transportation systems: a rule-based iterative design process", IEEE Transactions on Intelligent Transportation Systems **12**, 322–332 (2011).

[9] C. Dong, X. Ma, and B.-H. Wang, "Weighted congestion coefficient feedback in intelligent transportation systems", Physics Letters A - PHYS LETT A **374**, 1326–1331 (2010).

[10] J. Lee, Y.-S. Jeong, and J. Park, "S-itsf: a service based intelligent transportation system framework for smart accident management", Human-centric Computing and Information Sciences **5**, `10.1186/s13673-015-0054-x` (2015).

[11] I. Sumra, H. Hasbullah, J.-L. Ab Manan, M. Iftikhar, I. Ahmad, and A. Alghamdi, "A novel vehicular sms system (vss) approach for intelligent transport system (its)", in (Aug. 2011).

[12] M. Alsabaan, K. Naik, T. Khalifa, and S. Alaboodi, "Performance study of economical and environmentally friendly geocast routing in vehicular networks", IEEE Transactions on Vehicular Technology **64**, 3783–3789 (2015).

[13] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, *You only look once: unified, real-time object detection*, 2016.

[14] J. Redmon and A. Farhadi, *Yolov3: an incremental improvement*, 2018.

[15] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, *Yolov4: optimal speed and accuracy of object detection*, 2020.

[16] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple online and realtime tracking", 2016 IEEE International Conference on Image Processing (ICIP), 10.1109/icip.2016.7533003 (2016).

[17] P. K. Xingyi Zhou Dequan Wang, "Objects as points", (2019).

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

[19] 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 **111**, 98–136 (2015).

[20] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite", in 2012 ieee conference on computer vision and pattern recognition (IEEE, 2012), pp. 3354–3361.

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

[22] A. Milan, L. Leal-Taixé, I. D. Reid, S. Roth, and K. Schindler, "MOT16: A benchmark for multi-object tracking", CoRR **abs/1603.00831** (2016).

[23] J. Luiten, A. Osep, P. Dendorfer, P. H. S. Torr, A. Geiger, L. Leal-Taixé, and B. Leibe, "HOTA: A higher order metric for evaluating multi-object tracking", CoRR **abs/2009.07736** (2020).

[24] K. Bernardin and R. Stiefelhagen, "Evaluating multiple object tracking performance: the clear mot metrics", EURASIP Journal on Image and Video Processing **2008**, 1–10 (2008).

[25] E. Ristani, F. Solera, R. S. Zou, R. Cucchiara, and C. Tomasi, "Performance measures and a data set for multi-target, multi-camera tracking", CoRR **abs/1609.01775** (2016).