Efficient Online Transfer Learning for 3D Object Classification in Autonomous Driving

4 authors:

Rui Yang
Université de Technologie de Belfort-Montbéliard
2 PUBLICATIONS   1 CITATION

SEE PROFILE

Zhi Yan
Université de Technologie de Belfort-Montbéliard
59 PUBLICATIONS   1,171 CITATIONS

SEE PROFILE

Tao Yang
Northwestern Polytechnical University
16 PUBLICATIONS   251 CITATIONS

SEE PROFILE

Yassine Ruichek
Université de Technologie de Belfort-Montbéliard
269 PUBLICATIONS   2,374 CITATIONS

SEE PROFILE

Some of the authors of this publication are also working on these related projects:

Project   3L4AV: Lifelong learning of dynamic objects detection and tracking in adverse conditions for autonomous vehicles   View project

Project   Facial demographic estimation   View project

Efficient Online Transfer Learning for 3D Object Classification in Autonomous Driving

Rui Yang[1], *Student Member, IEEE*, Zhi Yan[1*], *Senior Member, IEEE*,
Tao Yang[2], Yassine Ruichek[1], *Senior Member, IEEE*

Abstract— Autonomous driving has achieved rapid development over the last few decades, including the machine perception as an important issue of it. Although object detection based on conventional cameras has achieved remarkable results in 2D/3D, non-visual sensors such as 3D LiDAR still have incomparable advantages in the accuracy of object position detection. However, the challenge also exists with the difficulty in properly interpreting point cloud generated by LiDAR. This paper presents a multi-modal-based online learning system for 3D LiDAR-based object classification in urban environments, including cars, cyclists and pedestrians. The proposed system aims to effectively transfer the mature detection capabilities based on visual sensors to the new model learning based on non-visual sensors through a multi-target tracker (i.e. using one sensor to train another). In particular, it integrates the Online Random Forests (ORF) [1] method, which inherently has the abilities of fast and multi-class learning. Through experiments, we show that our system is capable of learning a high-performance model for LiDAR-based 3D object classification on-the-fly, which is especially suitable for robotics in-situ deployment while responding to the widespread challenge of insufficient detector generalization capabilities.
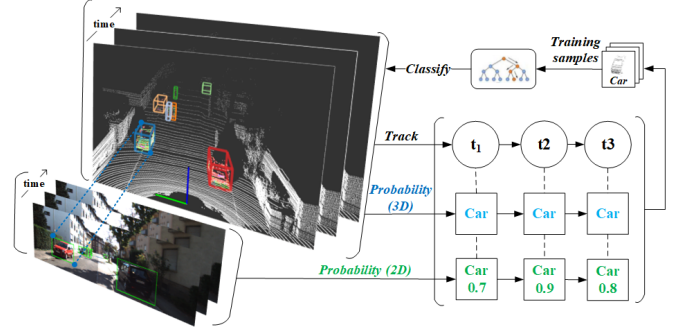
Fig. 1. Conceptual diagram of the proposed system. First, the output (2D bounding boxes shown in the lower left) of a high-performance 2D image object detector is used for annotation of 3D point cloud data (3D bounding boxes shown in the upper left). Then a multi-target tracker is used to correlate the clusters in the point cloud to generate a series of learning examples. Finally, a classifier is trained on-the-fly by using the Online Random Forests (ORF) [1].

I. INTRODUCTION

Object detection in 3D space is an essential portion for autonomous driving, which contains object localization and classification tasks. The main sensors currently used for this purpose include cameras [2], [3], [4] and LiDARs [5], [6], [7], while the latter is considered to be a strong support for the former, due to its high precision of range measurement and less sensitive to light conditions. The well-known case can be traced back to the 2007 DARPA Grand Challenge, in which the Stanford Junior robot car's primary sensor for obstacle detection such as pedestrians, signposts, and cars, is a 64-layer 3D LiDAR [8]. However, although highly anticipated, it is not straightforward to identify objects in the point cloud generated from such sensors, because some high recognition features such as texture and color are missing, making false positives more likely.

More challenging, in response to the changing environment [9], [10] and environment changes [11], [12], a feature-based model usually needs to be fine-tuned or even retrained to achieve acceptable performance over time or in new

[1]CIAD UMR 7533, Université Bourgogne Franche-Comté, UTBM, F-90010 Belfort, France. `firstname.lastname@utbm.fr`
[2]Unmanned System Research Institute, Northwestern Polytechnical University, China. `yangtao@nwpu.edu.cn`
*Corresponding Author: `zhi.yan@utbm.fr`

scenarios. An approach to eliminate this dilemma is to train a generalized model. However, the latter requires a large amount of labelled data, usually associated with high labor and machine costs. As for 3D LiDAR data, labelling sparse point clouds is tedious work and prone to human error, in particular if many variations of object pose, shape and size need to be correctly classified. Furthermore, timely learning of new samples to keep the model up-to-date is highly desirable to ensure the object detection performance of autonomous driving.

Compared with traditional offline learning (i.e. statistics), online learning is a paradigm more in line with the way humans learn. It usually only requires a few [13], [14] or even no [15] manually annotated samples to learn a high-performance model. Apart from this, the benefits it brings also include alleviating the computational burden when updating the model and improving the timeliness of the model, which is fully consistent with the development needs of autonomous driving for in-situ deployment [12] and long-term autonomy [9], [10], [16].

Based on today's popular multi-modal perception solutions, in this paper, we propose an efficient online transfer learning pipeline for 3D LiDAR-based object classification in autonomous driving. An intuitive graphical description is shown in Fig. 1. First, we extract object information from the 2D images, including the label associated with probability and the 2D bounding box. Meanwhile, the point cloud generated by the 3D LiDAR is segmented into clusters, and the 2D image-based object information is associated

with the clusters via the multi-sensor calibration and the synchronization of the sensory data. In other words, an image-based detector is used to automatically annotate the point cloud data to realize the knowledge transfer. Then, the clusters in different frames are correlated by a multi-target tracker, the probability of which type of object each track belongs to is calculated, and the high-confidence track is input into the Online Random Forests (ORF) [1] module as training samples. It is worth pointing out that, 1) the random forest (RF) is inherently fast and suitable for multi-class model learning, and 2) the efficiency of learning is reflected in the fact that only a part of the clusters in the track are labelled due to the limited visual scope of the camera, but the track usually contains a large number of samples for learning due to the large-range and long-distance sensing capabilities of the 3D LiDAR.

The contributions of this paper are twofold. First, we propose a novel implementation of the multi-sensor-based online learning framework [15] to allow efficient migration of mature image-based detection capabilities to 3D LiDAR, and use ORF to achieve rapid incremental learning. Second, for the first time, we apply our online learning framework to 3D LiDAR-based object classification including cars, cyclists and pedestrians, in urban environments for autonomous driving, and open source our code based on ROS [17] and Autoware [18] to the community: `https://github.com/epan-utbm/efficient_online_learning`.

II. Related Work

Online machine learning, with other closely related paradigm such as lifelong, transfer, continual, open-ended, and self-supervised learning, to name but a few, mainly refers to constantly update the knowledge model over time, and typically without human intervention. Pioneering work in this field can be traced back more than two decades [19], in which a lifelong learning perspective for mobile robot control is presented. With the rapid development of various related technologies including hardware and algorithms, in recent years, research on robotic online learning has become more and more extensive [13], [20], [14], [15], [21], [22]. In particular, Teichman and Thrun [14] presented a semi-supervised learning to the problem of track classification in 3D LiDAR data based on Expectation Maximization (EM) algorithm, which illustrated that learning of dynamic objects can benefit from tracking system. The proposed learning procedure starts with a small set of hand-labelled seed tracks and a large set of background tracks, while the latter is pre-collected in areas without any pedestrians, bicyclists, or cars. In contrast, our research further focuses on in-situ learning in the absence or with incomplete background knowledge.

On the other hand, perception systems based on multi-modal sensors are still the preferred solution for self-driving car developers, as there is no almighty and perfect sensor so far, and they all have limitations and edge cases [9]. In the past ten years, extensive research has been conducted on the use of multiple sensors for 3D object detection and tracking in the field of autonomous driving, and it is expected that more competitive performance can be obtained by effectively integrating the advantages of different sensors [5], [23]. As one of the main sensors, the camera is favored by academia and industry due to its relatively low cost and capacity of providing rich semantics about the environment. And with the rapid development of deep learning technology, image-based detection have shown convincing results [2], [3], [4], [24]. However, the performance of methods based on such passive sensors are subject to lighting conditions and the relatively slow detection speed, and it is difficult to provide accurate 3D position information of the object.

Although with relatively high cost, 3D LiDARs have been widely used for autonomous driving as they are capable of providing large-scale, long-distance and high-precision distance measurement without considering the lighting conditions. For example, Zhou and Tuzel [6] proposed an end-to-end trainable deep network architecture that transforms LiDAR point cloud data into dense tensors (called voxel) to enable GPU-accelerated computing. Alejandro et al. [25] explored the fusion of RGB and depth maps obtained by high-definition LiDAR detection and ranging in the multi-modal component using RF as a local expert to boost the accuracy. Nevertheless, these methods relies on considerable amount of manually annotated training data thus offline learned models to achieve effective object detection.

Through investigation of the state-of-the-art, none of the existing work in autonomous driving exploits information for multi-class learning from multi-sensor-based tracking in urban environments that our proposal does, that is, using one sensor to train another on-the-fly by fusing detections from both. Our work actually combines the advantages of multi-modal system diversity with the efficiency of semi-supervised learning, and integrates them into an online framework.

III. General Framework

In this section, we recall the general framework (illustrated in Fig. 2) of online transfer learning that was previously proposed in [15], which consists of four components: the *Stable Detector* and the *Learned Detector* are considered as the data entry, the *Target Tracker* synthesizes the detections for multiple targets tracking, and the *Label Generator* provides labelled training data for online learning of the classifier within the *Learned Detector*.

A. Stable Detector and Learned Detector

We assume two types of detectors in our general framework. The *Stable Detector* has good detection ability, which is typically pre-trained and provides a high level of confidence. The *Learned Detector* has no detection capability at the beginning, but it can continuously learn with the help of the *Stable Detector* over time. It is hoped that under this framework, the detection capabilities of the *Stable Detector* can be transferred to the *Learned Detector* on-the-fly. This mechanism is especially designed for those sensors whose data are difficult to annotate or features extracted from the data change greatly with the environment. From a machine
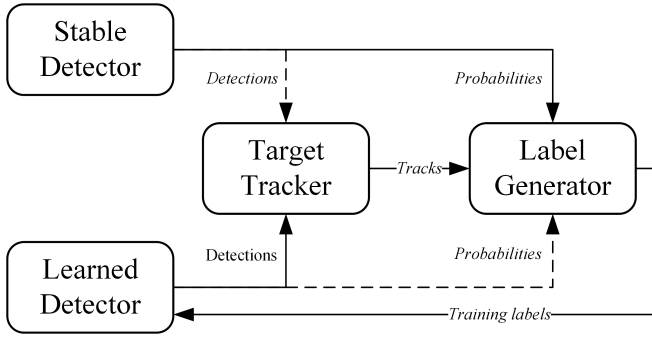
Fig. 2. Block diagram of the general framework.

learning perspective, the *Stable Detector* can provide labelled data, while the *Learned Detector* can provide both unlabelled and labelled data (as it is constantly learning). This shows that the evolution of the *Learned Detector* can be further regarded as semi-supervised learning to some extent.

B. Target Tracker and Label Generator

The *Target Tracker* plays a critical role in our framework. It correlates the detections from different detectors thereby improving the performance of object tracking on the one hand, and establishes a pipeline between the stable and the learned detectors thereby boosting the learning of the latter on the other hand. The *Label Generator* obtains tracks generated by the tracker and estimates the categories to which they belong, such as cars, pedestrians, cyclists, etc. Through the discrimination of the category of the entire track, all the samples on the track actually become learnable to the *Learned Detector*.

IV. EFFICIENT ONLINE TRANSFER LEARNING

In this section, we present our novel implementation as a specific instantiation of the general framework, with which we can test our system in autonomous driving. The detailed module diagram of our implementation is shown in Fig. 3. Specifically, the RGB image provided by the color camera is input to a pre-trained visual detector based on the state-of-the-art deep learning method [26] to obtain high-confidence label information of the sample. At the same time, the point cloud generated by the 3D LiDAR is segmented based on the Euclidean clustering algorithm [27] to form different clusters. Then, through the known extrinsic calibration parameters between the two sensors and the time stamp synchronization of different sensory data, the 3D bounding box information of the clusters is projected into the 2D space of the synchronized image frame, in order to match the visual detection to obtain the cluster labels and their corresponding probabilities. It is worth noting that the annotation result from this step is not the final label of the sample used for online learning. Instead, the clustering results and their annotations are first sent to the multi-target tracker to generate timely the tracks of the object, and then the discriminator will take the track and calculate the probability that the entire track belongs to a certain type of

Model	Car	Pedestrian	Cyclist	Runtime
YOLOv3 [28]	59.73%	38.51%	31.92%	21ms
Faster R-CNN [29]	81.57%	60.53%	57.83%	2635ms
EfficientDet-D1 [26]	75.83%	51.69%	46.92%	34ms

object, so that the final label of all samples on the track for online learning of the 3D LiDAR-based object classifier can be obtained.

Furthermore, our system integrates Autoware's [18] point cloud segmentation and multi-target tracking modules, that are based on traditional methods including Euclidean distance and Kalman filter rather than deep learning, because the former is more in line with Autoware's pursuit of industrial applications, especially for the purposes of trusted, explainable and certifiable. This idea is also one of the reasons why we are more willing to integrate traditional learning methods like SVM [15] and RF (in this paper) into our online framework. The details of each functional module are given below.

A. Visual Detector

The visual detector provides positional information of detected objects in 2D images, including the 2D bounding box coordinates, category and prediction score. It actually plays the role of "annotator" in our system, which is mainly used to guide the learning of the classifier in the 3D detector. In order for the visual detector to provide stable and accurate detection results, we tested three popular deep learning-based detection models including YOLOv3 [28], Faster R-CNN [29] and EfficientDet [26] on the KITTI dataset [30].

Table I summarizes the mAP (mean average precision) of different categories by calculating the arithmetic mean of the results of the three methods at different detection difficulties (i.e. moderate, easy and hard according to KITTI's benchmarks) and the corresponding runtime. It can be seem that Faster R-CNN and EfficientDet have competitive mAP, and both are better than YOLOv3. In terms of runtime, YOLOv3 achieved real-time performance while Faster R-CNN has a slower execution speed, meaning that the latter is currently not adaptive for autonomous driving scenarios. We finally chose EfficientDet-D1 to balance accuracy and real-time requirements, and filter out the output below a predetermined probability threshold[1] to prevent false detections.

B. 3D Detector

The 3D detector consists of three parts including a segmentor, a descriptor and a classifier. The 3D LiDAR scan is first segmented into different clusters. These unlabelled clusters will not be directly input into the descriptor for

---

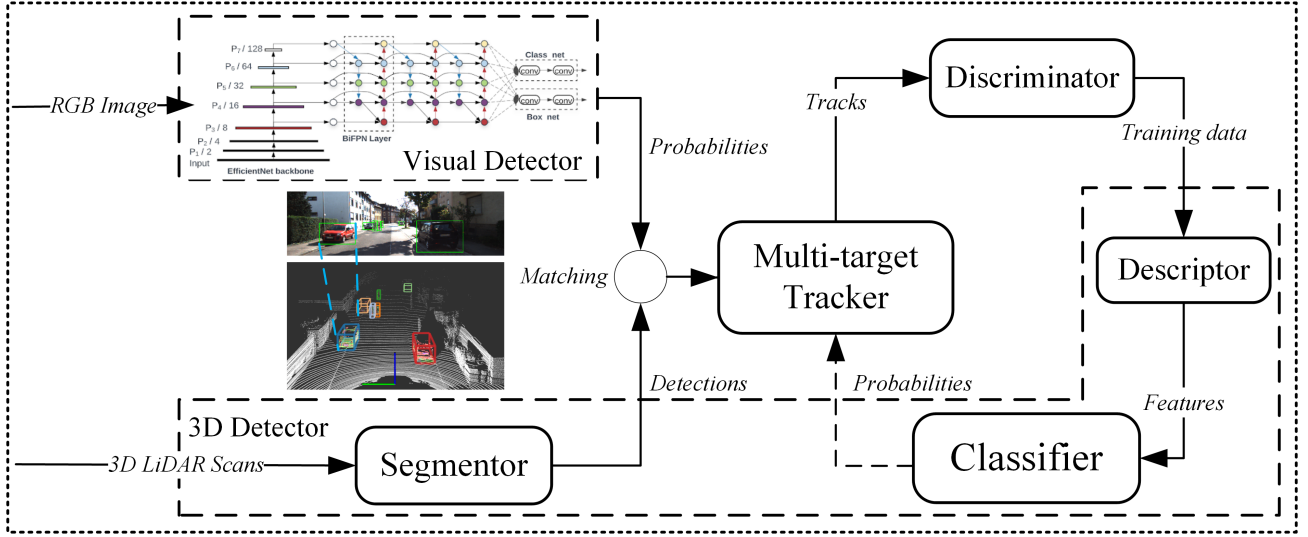[1]The threshold was set to 0.5 in our experiments.

Fig. 3. The detailed module diagram of multi-modal-based efficient online transfer learning system for object classification in autonomous driving.

feature extraction, but will be further processed by the multi-target tracker and discriminator before finally being used for online classifier training.

1) Segmentor: As input of this module, a 3D LiDAR scan is defined as a set of I points (i.e. a 3D point cloud):

$$P = \{p_i | p_i = (x_i, y_i, z_i) \in \mathbb{R}^3, i = 1, \ldots, I\} \quad (1)$$

Clusters are extracted from its 3D projection $P' = \lambda P$ where $\lambda = (1, 1, 0)$, based on the Euclidean distance [27] between points in 2D space. A cluster can then be defined as follows:

$$C_j \subset P', j = 1, \ldots, J \quad (2)$$

where J is the total number of clusters. A condition to avoid overlapping clusters is that they should not contain the same points, that is:

$$C_j \cap C_k = \emptyset, \text{ for } j \neq k, \text{ if } min\|p_j - p_k\|_2 \geq d \quad (3)$$

where the sets of points $p_j, p_k \in P'$ belong to the point clusters C_j and C_k respectively, if the minimum distance between p_j and p_k is greater than or equal to a given distance threshold d.

Before inputting the clusters as objectness detection to the tracker, two more processes are required, namely filtering and annotation. The former filters out clusters that are too large or too small, which means more attention will be paid to the objects that are more likely to be road participants. Using a predefined volumetric filter is effective, and can exclude most negative samples and background objects in our case:

$$\overline{C} = \{C_j | 0.1 \leq w_j \leq 5, \; 0.1 \leq d_j \leq 5, \; 0.3 \leq h_j \leq 5\} \quad (4)$$

where w_j resp. d_j resp. h_j represent the length resp. width resp. height (in meters) of the minimal cube containing C_j. On this basis, the efficiency of cluster-image matching and target tracking is also significantly improved.

Regarding the annotation, it is necessary to match the clustering results with the image detection provided by the

TABLE II

FEATURES FOR CLASSIFIER TRAINING

Feature	Description	Dimension
f_1	Number of points included in the cluster	1
f_2	Minimum cluster distance from the sensor	1
f_3	3D covariance matrix of the cluster	6
f_4	Normalized moment of inertia tensor	6
f_5	Slice feature for the cluster	20
f_6	Reflection intensity distribution	27

visual detector in order to obtain a pre-label of the clusters. Specifically, we project the 3D bounding box of the clusters onto the 2D image, and determine whether they are the same object by measuring the IoU (Intersection over Union)[2].

2) Descriptor: Six features (illustrated in Table II) are extracted from the clusters by the descriptor for later classifier training. The set of feature values of each sample C_j forms a vector $f_j = (f_1, \ldots, f_6)$. Features from f_1 to f_4 were introduced by [31], while features f_5 and f_6 were proposed by [7]. The selection of these features is mainly based on the needs of real-time perception and lightweight on-board computing for autonomous driving. For the performance comparison of each feature, please refer to [7], [13], [31].

3) Classifier: RF is an ensemble learning method for classification, regression and other tasks that operate by constructing a multitude of decision trees. Compared with the SVM used in our previous work [15], RF has inherent advantages of speed and multi-classification. It also natively supports incremental learning and is conducive to the preservation of knowledge. The classic RF algorithm uses batch training and generates a series of decision trees based on global data, which cannot be straightforwardly applied to online situations. A proven online RF method (ORF) [1] referencing the idea of online-bagging, makes full use of the

---

[2]In our experiments, we required an IoU of 0.7 for cars, and an IoU of 0.5 for pedestrians and cyclists, as per KITTI.

fast and strong generalization capabilities of RFs, has been adopted in our framework. Later experiments in this paper will show that with the increase in the number of learning samples, the performance of online learned classifier will gradually approach the offline learned one.

Specifically, the statistics in ORF are gathered over time, and the decision when to split on a tree depends on 1) the minimum number of samples (α) of a node need to reach before splitting, and 2) the minimum gain (β) a split has to achieve. Thus, a node splits when $|R_j| > \alpha$ and $\exists s \in S : \triangle L(R_j, s) > \beta$, where $\triangle L(R_j, s)$ denotes the gain with respect to a test s in node j which can be measured as:

$$\triangle L(R_j, s) = L(R_j) - \frac{|R_{jls}|}{|R_j|} L(R_{jls}) - \frac{|R_{jrs}|}{|R_j|} L(R_{jrs}) \quad (5)$$

where R_{jls} and R_{jrs} are respectively the left and right partitions made by the test s, $|...|$ denotes the number of samples in each partition, and $\triangle L(R_j, s) \geq 0$. Tests with higher gains will lead to better data splitting in terms of reducing node impurities. Therefore, when splitting a node, the test with the highest gain is selected as the main decision for the node.

On the basis of the original algorithm, we added support for data streams and improved the functions of small batch sample learning and real-time model storage. The improved version supports incremental learning with a single or a small number of samples. For more details, please refer to our released code.

C. Multi-target Tracker

The multi-target tracker extracted from Autoware is based on the IMM-UKF-PDA framework [32], [33]. Data association and state estimation are two key issues in target tracking [34], which are even more challenging in autonomous driving. The idea of the PDA (Probabilistic Data Association) algorithm is that the update state of the target is the weighted sum of the probabilities of each possible target state update. UKF (Unscented Kalman Filter) is a combination of Unscented Transform (UT) and standard Kalman filter. Through the UT transformation, the nonlinear system equation can be adapted to the standard Kalman filter system under the linear assumption. This can effectively overcome the problems of low estimation accuracy and poor stability, therefore provide robust tracking performance.

The IMM (Interactive Multiple Model) algorithm allows to efficiently manage multiple filter models. Assume that the state equation and measurement equation of the target are as follows:

$$\begin{cases} x_{k+1} &= f_{x_k, m_k} + \omega_{k, m_k} \\ z_k &= h_{x_k, m_k} + v_{k, m_k} \end{cases} \quad (6)$$

where m_k is the effective mode at sampling time k. Let the system model set be $M = (m_1, m_2, \ldots, m_n)$, where the conversion process conforms to the Markov chain. Under unconditional constraints, the transition probability $P(m_i(k+1)|m_j(k))$ from $m_j(k)$ to $m_i(k+1)$ is denoted as π_{ji}. The probability that model m_i is a matching model

at time k is called model probability $P(m_i(k)|z_k)$, which is denoted as $\mu_i(k)$, where $Z = (z_1, z_2, \ldots, z_k)$ represents the measurement set of the system. The initial Markov transition probability satisfies the condition:

$$\begin{cases} \pi_{ji} = P(m_i(k)|m_j(k-1)), m_i, m_j \in M \\ \sum_{i=1}^{n} \pi_{ji} = 1, j = 1, \ldots, n \end{cases} \quad (7)$$

and the model prediction probability is defined as:

$$\mu_{j|i}(k-1) = \frac{\pi_{ji}\mu_j(k-1)}{\sum_{j=1}^{n} \pi_{ji}\mu_j(k-1)} \quad (8)$$

Overall, the IMM-UKF filtering process includes input interaction, UKF filtering, model probability update and output fusion. For more details, please refer to [32], [33].

D. Discriminator

The discriminator generates training samples with label by measuring the *track probability* [15] based on Bayes' theorem. The idea is to measure the likelihood that a track belongs to a specified object such as cars, pedestrians, cyclists, etc., which is defined as follows. Let $P(y_i|x_i, d_j)$ denote the probability that example x_i is an object with its category label y_i predicted by detector $d_j \in \mathcal{D}$ at the precise time t, then the track probability $P(Y_T|X_T, \mathcal{D})$ is computed by integrating the predictions of the different detectors according to the following formula:

$$P(Y_T|X_T, \mathcal{D}) = \frac{odds_{X_T}}{1 + odds_{X_T}} \quad (9)$$

where

$$odds_{X_T} = \prod_{i=1}^{t} \prod_{j=1}^{K} odds_{x_i}^j \quad (10)$$

and

$$odds_{x_i}^j = \frac{P(y_i|x_i, d_j)}{1 - P(y_i|x_i, d_j)} \quad (11)$$

By thresholding[3] $P(Y_T|X_T, \mathcal{D})$, high-confidence samples are then sent to the descriptor for feature extraction.

V. EVALUATION

In this section, we first illustrate our experimental setup, then evaluate individually the performance of the ORF module followed by the entire system, with further analysis of the system architecture through ablation experiments, and finally give qualitative results and discuss the advantages and limitations of our proposed system.

A. Experimental Setup

The experiments were conducted on the KITTI dataset [30], which was collected in urban environments with a car equipped with various sensors including several color and gray-scale cameras and a Velodyne 64-layer LiDAR. On the one hand, the 7481 training frames (with ground-truth annotations) provided in the 3D object detection benchmark were split into the ratio of 6:2:2 for training, validation and testing respectively. The image data

---

[3]The threshold was set to 0.7 in our experiments.

was used to evaluate EfficientDet and other deep learning based methods (c.f. Table I) in 2D image detection by 5-fold cross validation, while the point cloud data was used to evaluate the performance of ORF-based classifier (c.f. Sec. V-B).

On the other hand, considering the significance of temporally continuous scenes (i.e. time-adjacent frames) to the online learning framework, the raw data (five categories including city, residential, road, campus, and person) provided by KITTI were used to online train the 3D LiDAR-based classifier (c.f. Sec. V-C), which does not contain any annotations in the image or point cloud. Additionally, scene fragments that last longer than 60 seconds were excluded, since most of them are used for visual odometry task, which means that the scene is typically monotonous and the objects contained are usually stationary vehicles. Moreover, an under-sampling was applied to process training samples in order to reduce the negative effect of unbalanced distribution of learning samples (especially with many cars and few cyclists) on the performance of the learned classifier.

Our framework has been fully implemented into ROS with very high modularity. All components are ready for download and used by the community. All the experiments reported in this paper were performed with Ubuntu 18.04 LTS (64-bit) and ROS Melodic, with an Intel i9-9900K CPU and 64GB RAM, and a NVIDIA GeForce RTX 2080 GPU with 8GB RAM.

B. ORF Performance

In order to evaluate the performance of the ORF module, we randomly selected 1000 samples from our training subset to form a data stream as the input of ORF. Specifically, every time the classifier learns 100 samples, we will save a model locally and evaluate it on our test subset to report as the result of an iteration. We report the results of the first ten iterations. In addition, a classic batch RF-based classifier [35] offline trained with the same 1000 samples is served as the baseline for comparison. Regarding the structure of the forest, both online and offline manners are consistent in parameter settings, i.e., $trees = 100, depth = 50, epochs = 20, split\_threshold = 50$, while the other parameters remain the default values as in the released code.

The experimental results are shown in Fig. 4. F1-score is used as the evaluation metric, including the average accuracy (ACC, numerically equal to micro-F1-score) and the macro average (MaA, i.e. macro-F1-score). It can be seen that as the number of training samples increases, the performance of the online learned classifier quickly approaches the offline trained one.

C. System Performance

In order to evaluate the performance of our proposed system, we first compare the classification results of the 3D LiDAR-based classifier learned online with KITTI's raw data under our framework (c.f. Fig. 3), with the independently learned (with ground-truth annotations) ORF-based classifier reported in the previous section. We report the results after
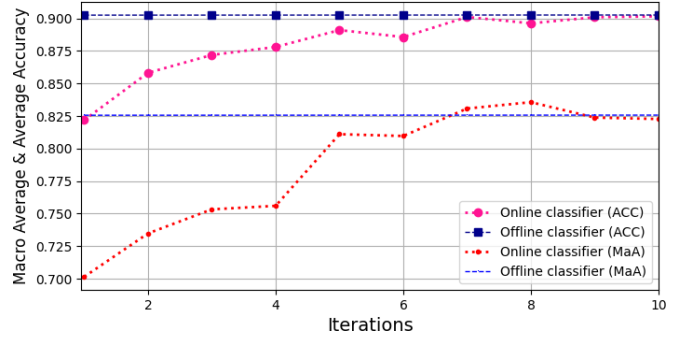

Fig. 4. Performance comparison of the ORF-based online learned and the RF-based offline trained 3D LiDAR object classifiers.

the final iteration (i.e. after learning 1000 samples), as shown in Fig. 5. Two confusion matrices containing three categories (i.e. car, pedestrian and cyclist) are used to evaluate the performance of the two classifiers. The abscissa indicates the predicted result while the ordinate represents the true label. Different color depths correspond to proportions, and the darker the color, the higher the percentage of correct classification.

The results show that although the performance of the 3D LiDAR-based classifier learned under our proposed system cannot surpass the ORF-only classifier with limited samples, there is no huge gap between the two classifiers. Since the proportion of vehicles in the test subset is the highest, it is reasonable that fewer vehicles are incorrectly classified into other categories. There are relatively more incorrect classifications of pedestrians and cyclists because the segmentor module in our system suffers from difficulties of segmenting people or bicycle groups, so that the classifier learns false positive samples.

Furthermore, we analyze and verify our system architecture through ablation experiments. It can be seen from Fig. 6 that the performance of the classifier learned without the visual detector but only relying on a volumetric template cannot be compared with other results. This is because using only the size of the cluster to determine its category produces considerable number of false positive samples. The classifier under the system without the target tracker learns better in simple scenarios (i.e. the first 1500 frames), but the learning efficiency decreases as the scenarios become more complex (i.e. more pedestrians and cyclists appeared in 1500-2000 frames and 2500-3000 frames). Instead, the classifier learned under the complete system shows the best performance.

D. Qualitative Analysis

We mainly focus on the qualitative analysis of three aspects, as shown in Fig. 7. The first is about the segmentation of the point cloud. Overall, the segmentor shows good performance in both simple and complex scenes, but when facing objects that are too close (forming a group), there is still room for improvement. For example, in the area shown in the red box in the lower half of Fig. 7, people and benches are clustered as a whole. The second is the performance of the visual detector that plays the role of the trainer. We can
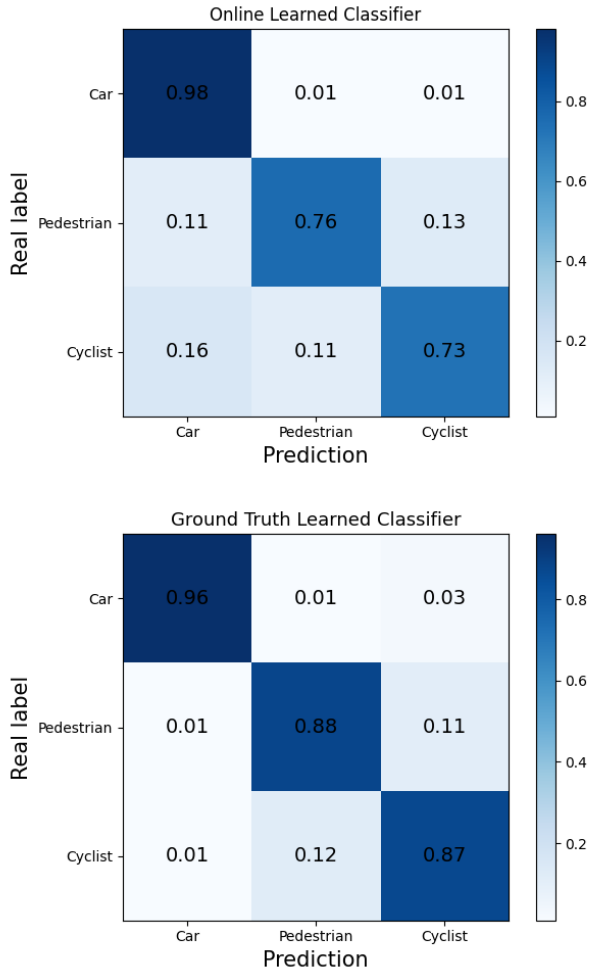
Fig. 5. Confusion matrices of online learned 3D LiDAR-based classifiers respectively under our proposed system (with raw data) and using ORF alone (with ground-truth annotations).

see that image detection still shows acceptable performance in complex environments, and the matching with point cloud clusters is also robust. The third point is about the role of the multi-target tracker. It can be seen from the upper part of Fig. 7 that although the cyclist has left the camera's visual range, the classifier can still learn the sample of the object in the point cloud through the association of the tracker (indicated by the blue box). Similar situations are shown in the lower half of Fig. 7. This undoubtedly greatly improves the learning efficiency of the entire system.

VI. CONCLUSION

In this paper, we presented a novel implementation of the previously proposed multi-sensor online transfer learning framework, which is capable of efficiently learning a 3D LiDAR-based multi-class classifier on-the-fly from a monocular camera-based detector in urban environments. Specifically, we explored how to use today's popular deep learning technology to help learn from data that is not easy to interpret, such as sparse point clouds. To do so, leveraging a powerful multi-target tracker provided by Autoware,
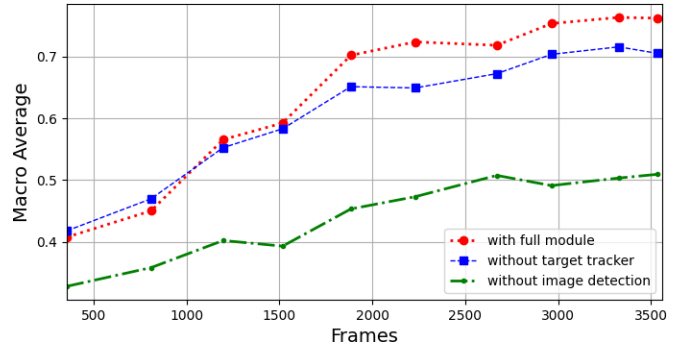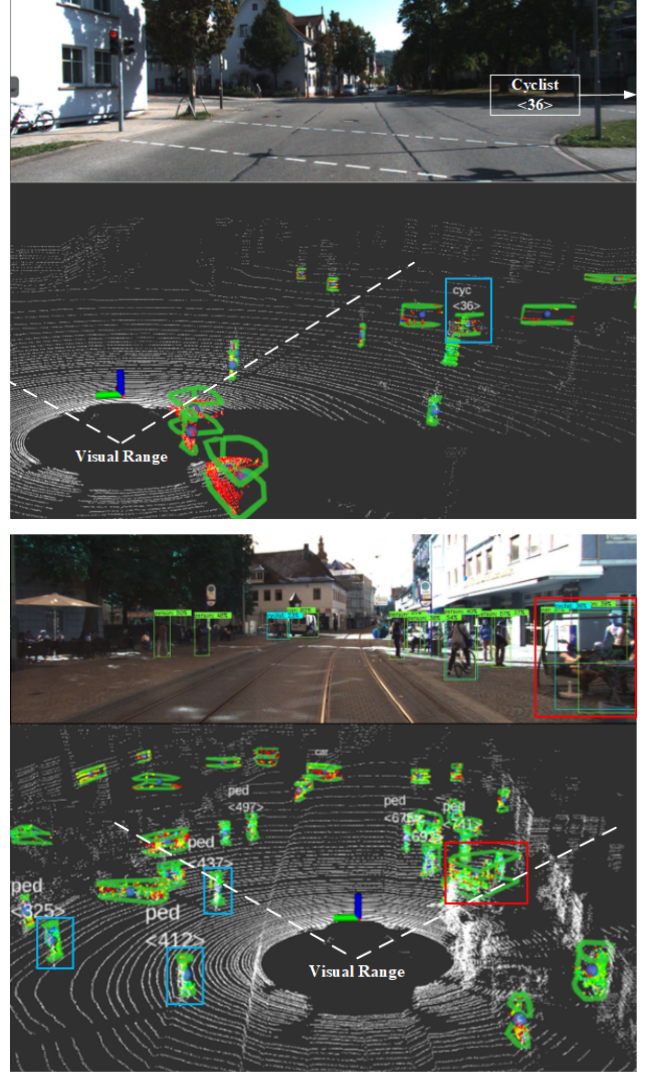

Fig. 6. Results of ablation experiments.

Fig. 7. Two pairs of synchronized frames include the color image and its corresponding 3D LiDAR scan. The coordinate axis in the point cloud represents the position of the 3D LIDAR sensor. The colored point cloud enclosed by the bounding box represents the samples input to the classifier. Note that only those labelled samples will be learned by the classifier, such as the cyclist in the upper iamge. The upper and lower images respectively show the performance of our system in simple and complex scenarios.

we actually established a pipeline between the pre-trained detector and the learning detector to achieve knowledge

transfer, and used ORF to realize rapid incremental learning. A very promising feature of the proposed solution is that a new object classifier can be learned and updated directly from the deployment environment, thus getting rid of the dependence of model learning on manual data annotation on the one hand, and improving the system's ability to adapt to environmental changes on the other hand.

For the first time, we deployed our online learning framework in the field of autonomous driving and demonstrated promising results through experiments. The proposed system has been fully implemented into ROS with a high level of modularity, and open sourced to the community.

Despite the encouraging results, there are still several aspects that can be improved. Future work will include improving the quality of online learning samples thus the classifier performance, and experimenting with some other datasets (for different environments) as well as our own self-driving cars equipped with low-power computing units.

ACKNOWLEDGMENT

REFERENCES

[1] A. Saffari, C. Leistner, J. Santner, M. Godec, and H. Bischof, "On-line random forests," in *ICCV Workshops*, 2009, pp. 1393–1400.

[2] X. Chen, K. Kundu, Z. Zhang, H. Ma, S. Fidler, and R. Urtasun, "Monocular 3d object detection for autonomous driving," in *Proceedings of CVPR*, 2016, pp. 2147–2156.

[3] A. Mousavian, D. Anguelov, J. Flynn, and J. Kosecka, "3d bounding box estimation using deep learning and geometry," in *Proceedings of CVPR*, 2017, pp. 7074–7082.

[4] B. Xu and Z. Chen, "Multi-level fusion based 3d object detection from monocular images," in *Proceedings of CVPR*, 2018, pp. 2345–2353.

[5] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3d object detection network for autonomous driving," in *Proceedings of CVPR*, 2017, pp. 1907–1915.

[6] Y. Zhou and O. Tuzel, "Voxelnet: End-to-end learning for point cloud based 3d object detection," in *Proceedings of CVPR*, 2018, pp. 4490–4499.

[7] K. Kidono, T. Miyasaka, A. Watanabe, T. Naito, and J. Miura, "Pedestrian recognition using high-definition LIDAR," in *Proceedings of IV*, 2011, pp. 405–410.

[8] M. Montemerlo, J. Becker, S. Bhat, H. Dahlkamp, D. Dolgov, S. Ettinger, D. Haehnel, T. Hilden, G. Hoffmann, B. Huhnke, *et al.*, "Junior: The stanford entry in the urban challenge," *Journal of Field Robotics*, vol. 25, no. 9, pp. 569–597, 2008.

[9] Z. Yan, L. Sun, T. Krajník, and Y. Ruichek, "EU long-term dataset with multiple sensors for autonomous driving," in *Proceedings of IROS*, 2020, pp. 10 697–10 704.

[10] T. Krajník, J. P. Fentanes, J. M. Santos, and T. Duckett, "Fremen: Frequency map enhancement for long-term mobile robot autonomy in changing environments," *IEEE Transactions on Robotics*, vol. 33, no. 4, pp. 964–977, 2017.

[11] Z. Yan, S. Schreiberhuber, G. Halmetschlager, T. Duckett, M. Vincze, and N. Bellotto, "Robot perception of static and dynamic objects with an autonomous floor scrubber," *Intelligent Service Robotics*, vol. 13, no. 3, pp. 403–417, 2020.

[12] L. Sun, Z. Yan, S. M. Mellado, M. Hanheide, and T. Duckett, "3DOF pedestrian trajectory prediction learned from long-term autonomous mobile robot deployment data," in *Proceedings of ICRA*, 2018, pp. 1–7.

[13] Z. Yan, T. Duckett, and N. Bellotto, "Online learning for 3d lidar-based human detection: experimental analysis of point cloud clustering and classification methods," *Autonomous Robots*, vol. 44, no. 2, pp. 147–164, 2020.

[14] A. Teichman and S. Thrun, "Tracking-based semi-supervised learning," in *Proceedings of RSS*, 2011.

[15] Z. Yan, L. Sun, T. Duckett, and N. Bellotto, "Multisensor online transfer learning for 3d lidar-based human detection with a mobile robot," in *Proceedings of IROS*, 2018, pp. 7635–7640.

[16] L. Kunze, N. Hawes, T. Duckett, and M. Hanheide, "Introduction to the special issue on AI for long-term autonomy," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 4431–4434, Oct. 2018.

[17] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: an open-source robot operating system," in *ICRA Workshop on Open Source Software*, 2009.

[18] S. Kato, S. Tokunaga, Y. Maruyama, S. Maeda, M. Hirabayashi, Y. Kitsukawa, A. Monrroy, T. Ando, Y. Fujii, and T. Azumi, "Autoware on board: enabling autonomous vehicles with embedded systems," in *Proceedings of ICCPS*, 2018, pp. 287–296.

[19] S. Thrun, "A lifelong learning perspective for mobile robot control," in *Proceedings of IROS*, 1994, pp. 23–30.

[20] Z. Yan, T. Duckett, and N. Bellotto, "Online learning for human classification in 3d lidar-based tracking," in *Proceedings of IROS*, 2017, pp. 864–871.

[21] F. Majer, Z. Yan, G. Broughton, Y. Ruichek, and T. Krajnik, "Learning to see through haze: Radar-based human detection for adverse weather conditions," in *Proceedings of ECMR*, 2019, pp. 1–6.

[22] G. Broughton, F. Majer, T. Rouček, Y. Ruichek, Z. Yan, and T. Krajník, "Learning to see through the haze: Multi-sensor learning-fusion system for vulnerable traffic participant detection in fog," *Robotics and Autonomous Systems*, p. 103687, 2020.

[23] Z. Wang, W. Zhan, and M. Tomizuka, "Fusing bird's eye view lidar point cloud and front view camera image for 3d object detection," in *Proceedings of IV*, 2018, pp. 1–6.

[24] P. Li, H. Zhao, P. Liu, and F. Cao, "Rtm3d: Real-time monocular 3d detection from object keypoints for autonomous driving," *arXiv preprint arXiv:2001.03343*, vol. 2, 2020.

[25] A. González, D. Vázquez, A. M. López, and J. Amores, "On-board object detection: Multicue, multimodal, and multiview random forest of local experts," *IEEE Transactions on Cybernetics*, vol. 47, no. 11, pp. 3980–3990, 2016.

[26] M. Tan, R. Pang, and Q. V. Le, "Efficientdet: Scalable and efficient object detection," in *Proceedings of CVPR*, 2020, pp. 10 781–10 790.

[27] R. B. Rusu, "Semantic 3D object maps for everyday manipulation in human living environments," Ph.D. dissertation, Computer Science department, Technische Universitaet Muenchen, Germany, 2009.

[28] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[29] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *arXiv preprint arXiv:1506.01497*, 2015.

[30] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Proceedings of CVPR*, 2012.

[31] L. E. Navarro-Serment, C. Mertz, and M. Hebert, "Pedestrian detection and tracking using three-dimensional ladar data," in *Proceedings of FSR*, 2009, pp. 103–112.

[32] A. A. Rachman, "3d-lidar multi object tracking for autonomous driving: multi-target detection and tracking under urban road uncertainties," Ph.D. dissertation, Delft University of Technology, November 2017.

[33] M. Schreier, "Bayesian environment representation, prediction, and criticality assessment for driver assistance systems," *at-Automatisierungstechnik*, vol. 65, no. 2, pp. 151–152, 2017.

[34] N. Bellotto, S. Cosar, and Z. Yan, "Human detection and tracking," in *Encyclopedia of Robotics*, M. H. Ang, O. Khatib, and B. Siciliano, Eds. Springer, 2018, pp. 1–10.

[35] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.