



**HAL**  
open science

# Online continual learning for 3D detection of road participants in autonomous driving

Rui Yang

► **To cite this version:**

Rui Yang. Online continual learning for 3D detection of road participants in autonomous driving. Other. Université Bourgogne Franche-Comté, 2023. English. NNT : 2023UBFCA021 . tel-04587549

**HAL Id: tel-04587549**

**<https://theses.hal.science/tel-04587549v1>**

Submitted on 24 May 2024

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

**THÈSE DE DOCTORAT**  
**DE L'ÉTABLISSEMENT UNIVERSITÉ BOURGOGNE-FRANCHE-COMTÉ**  
**PRÉPARÉE À L'UNIVERSITÉ DE TECHNOLOGIE DE BELFORT-MONTBÉLIARD**

École doctorale n°37  
Sciences Pour l'Ingénieur et Microtechniques

Doctorat d'Informatique

par

RUI YANG

Online Continual Learning for 3D Detection of Road Participants in  
Autonomous Driving

Thèse présentée et soutenue à Montbéliard, le 13 Décembre 2023

Composition du Jury :

M. SNOUSSI HICHEM	Professeur à l'Université de technologie de Troyes	Président
M. SIDIBE DÉsirÉ	Professeur à l'Université d'Evry-Paris Saclay	Rapporteur
M. KRAJNIK TOMAS	Professeur à l'Université technique de Prague	Rapporteur
M. RUICHEK YASSINE	Professeur à l'UBFC	Directeur de thèse
M. YAN ZHI	Maître de conférences à l'UBFC	Codirecteur de thèse



# ACKNOWLEDGEMENTS

First and foremost, I would like to express my sincere gratitude to my dissertation supervisors, Professor Yassine Ruichek and Dr. Zhi Yan. Their expertise and mentorship have been instrumental in shaping my research efforts. This dissertation would not have been possible without their unwavering support and guidance.

Professor Yassine Ruichek is a cheerful, meticulous, and insightful scholar. I will always remember the firm handshake we exchanged when we first met, which marked the first lesson I learned from him. To Dr. Zhi Yan, I am deeply thankful for his endless patience, steadfast support in both my academic and personal endeavors, and also the invaluable experiences of debugging together. Beyond research, I have learned from him the attitude towards scholarship and the principles of conduct.

I must also acknowledge the opportunity to study in France and the financial support provided by the China Scholarship Council. Without these, everything would have remained a distant dream. Furthermore, I extend my heartfelt thanks to my colleagues at EPAN in UTBM for their assistance, enthusiasm, and emotional support. They are the ones who remind me that life is more than just data and algorithms, but also the caffeine and dessert offered by them that fuels my day and late-night coding sessions.

Last but not least, to my dear friends and family, thank you for tolerating my erratic schedule, occasional existential crises, and the constant struggle with time zones. Thank you for always believing in me, even when I didn't. Your love and encouragement have kept me sane and grounded.

To those I haven't mentioned, please know that your contributions have not gone unnoticed. Whether it was a word of encouragement, a listening ear, or just a well-timed meme, every bit of support has been deeply appreciated.



# CONTENTS

<b>I</b>	<b>Context and Problems</b>	<b>1</b>
<b>1</b>	<b>Introduction</b>	<b>3</b>
1.1	Background and Motivation . . . . .	3
1.2	Challenges . . . . .	6
1.2.1	Generation of Samples . . . . .	6
1.2.2	Preservation of Knowledge . . . . .	6
1.2.3	Avoidance of Catastrophic Forgetting . . . . .	6
1.3	Objectives . . . . .	7
1.4	Thesis Organization . . . . .	9
<b>2</b>	<b>Related Work</b>	<b>11</b>
2.1	Introduction . . . . .	11
2.2	Upstream Tasks . . . . .	11
2.2.1	Online Learning . . . . .	12
2.2.1.1	Challenges . . . . .	12
2.2.1.2	Approaches . . . . .	13
2.2.2	Continual Learning and Lifelong Learning . . . . .	17
2.2.2.1	Challenges . . . . .	19
2.2.2.2	Approaches . . . . .	20
2.3	Downstream Tasks . . . . .	31
2.3.1	3D Detection of Road Participants . . . . .	31
2.4	Discussion . . . . .	35

<b>II</b>	<b>Contribution</b>	<b>39</b>
<b>3</b>	<b>Preliminary Knowledge</b>	<b>41</b>
3.1	The Methodology . . . . .	42
3.2	Data Used for Experiments . . . . .	45
3.2.1	Open Datasets . . . . .	45
3.2.2	Data Selection . . . . .	46
3.3	Evaluation Metrics . . . . .	47
3.3.1	Classification Performance Evaluation . . . . .	47
3.3.2	Detection Performance Evaluation . . . . .	50
3.4	Conclusion . . . . .	52
<b>4</b>	<b>Efficient Sample Generation for Online Learning</b>	<b>53</b>
4.1	Introduction . . . . .	54
4.2	Problem Formulation . . . . .	56
4.2.1	Online Transfer Learning . . . . .	56
4.2.2	Information Fusion . . . . .	57
4.3	General Framework . . . . .	57
4.4	Implementation . . . . .	60
4.4.1	Off-the-shelf Detector . . . . .	60
4.4.2	Learn-by-use Detector . . . . .	61
4.4.2.1	Ground Filtering . . . . .	62
4.4.2.2	Clustering . . . . .	63
4.4.2.3	Feature Extraction . . . . .	64
4.4.2.4	Online Classifier . . . . .	65
4.4.3	Multi-target Tracker . . . . .	65
4.4.3.1	Spatial Association . . . . .	66
4.4.3.2	Temporal Association . . . . .	66
4.4.4	Sample Annotator . . . . .	68
4.5	Experimental Results . . . . .	69
4.5.1	Detection of Road Participants . . . . .	69

4.5.2	Comparison with Other Methods . . . . .	71
4.5.3	Size Estimation . . . . .	73
4.5.4	Ablation Study . . . . .	73
4.6	Qualitative Analysis . . . . .	74
4.7	Conclusion . . . . .	77
<b>5</b>	<b>Efficient Online Model Training and Access</b>	<b>79</b>
5.1	Introduction . . . . .	80
5.2	Problem Formulation . . . . .	81
5.2.1	Efficient Model Training . . . . .	81
5.2.2	Efficient Model Access . . . . .	81
5.3	Methodology . . . . .	81
5.3.1	Online Random Forest . . . . .	82
5.3.2	Few-shot Training . . . . .	82
5.3.3	Octree . . . . .	83
5.4	Experimental Results . . . . .	84
5.4.1	Online Random Forest vs. Random Forest . . . . .	85
5.4.2	Integrated-ORF vs. Standalone-ORF . . . . .	86
5.4.3	Octree vs. Binary Tree . . . . .	87
5.4.4	Learn on KITTI Test on Waymo . . . . .	88
5.5	Discussion . . . . .	89
5.5.1	Online Adaptability . . . . .	89
5.5.2	Convergence and stability analysis . . . . .	90
5.6	Conclusion . . . . .	91
<b>6</b>	<b>Overcoming Catastrophic Forgetting in Online Learning</b>	<b>93</b>
6.1	Introduction . . . . .	94
6.2	Problem Formulation . . . . .	97
6.3	Long-Short-Term Online Learning . . . . .	98
6.4	Implementation . . . . .	99
6.4.1	Learning Sample . . . . .	99



6.4.2	Short-term Learning . . . . .	101
6.4.3	Long-term Control . . . . .	101
6.4.3.1	Information Gatherer . . . . .	101
6.4.3.2	Dynamic Gate Controller . . . . .	102
6.4.3.3	Weight Allocator . . . . .	103
6.5	Experimental Results . . . . .	104
6.5.1	Experimental Setups . . . . .	104
6.5.1.1	Datasets . . . . .	105
6.5.1.2	Comparison Models . . . . .	105
6.5.1.3	Implementation details . . . . .	106
6.5.2	Evaluation across Datasets . . . . .	106
6.5.3	LSTOL vs. EOTL . . . . .	108
6.5.4	Short-term Learner Assessment . . . . .	110
6.6	Conclusion . . . . .	111
<b>III</b>	<b>Conclusion</b>	<b>113</b>
<b>7</b>	<b>General conclusion</b>	<b>115</b>
7.1	Conclusions . . . . .	115
7.2	Future Research Directions . . . . .	117
<b>A</b>	<b>Publications</b>	<b>137</b>
A.1	Conferences . . . . .	137
A.2	Journals . . . . .	137

# I

## CONTEXT AND PROBLEMS



# INTRODUCTION

## 1.1/ BACKGROUND AND MOTIVATION

The field of autonomous driving has experienced remarkable advancements in recent years, with the aim of creating safer and more efficient transportation systems. A crucial component of autonomous vehicles is their ability to perceive and understand the surrounding environment. Object detection play pivotal roles in this process, enabling the vehicle to identify and monitor objects such as pedestrians, vehicles, or obstacles. However, autonomous vehicles will serve as daily transportation in human life, and the environments in which they will be deployed are often complex and dynamic. In addition, they are exposed to various weather conditions, changing lighting conditions, perception occlusion, and diverse interactions with other road participants. These factors pose structural challenges to the accuracy, robustness, and adaptability of object detection systems for autonomous vehicles.

On the one hand, from the hardware level, camera-based object detection has shown convincing results in both academia and industry. However, solutions based on pure visual perception are currently unable to meet the needs of autonomous driving, especially when the issue involves driving safety. Therefore, other modalities of sensors are integrated into autonomous vehicles, especially radar and LiDAR. Among them, 3D LiDAR has become the de facto standard configuration because it can provide more accurate object distance information than cameras (Qian et al., 2022). In addition, as this sensor can also provide a rough geometry of an object, coupled with its insensitivity to lighting conditions, researchers are interested in implementing object detection based on the data it provides, namely point cloud. However, due to the lack of easy-to-learn color and texture information, there is still a significant gap between the current object detection performance based on 3D LiDAR and that based on cameras.

On the other hand, from a software level, deep learning methods, particularly Convolutional Neural Network (CNN), achieved state-of-the-art (SOTA) performance in various

object detection tasks. Single-shot detectors such as You Only Look Once (YOLO) (Redmon and Farhadi, 2018) or two-shot detectors such as Faster R-CNN (Ren et al., 2015) have proven effective for real-time camera-based object detection. PointNet (Qi et al., 2017a) and its subsequent methods (Qi et al., 2017b, 2018) show the promising performance of CNN in point cloud data processing. However, these models are typically trained offline on large and fixed datasets, and their performance may degrade when encountering new and previously unseen data during deployment. To address these challenges and improve the overall robustness and reliability of autonomous driving systems, there is a growing interest in applying Online Learning methods to object detection (Kuznetsova et al., 2015; Perez-Rua et al., 2020; Wang et al., 2021).

Online Learning (OL) (Hoi et al., 2021), originally developed in the machine learning community, driven by the high cost of batch training and the lack of adaptability of the model, aims to learn from ordered (such as time series) data and update the model in real time without saving any learned data. The updated model is then used to make predictions on future (learning) data. It should be emphasized that OL is a model training method rather than a model. OL is particularly well-suited for deployment in the field of autonomous driving, and even more broadly, robotics, because:

1. The operating environment for agents is typically dynamic (Krajník et al., 2017; Yan et al., 2020c; Sun et al., 2018), and usually changing unpredictably. This necessitates the online adaptability for the model carried by the agent, enabling it to adjust to unforeseen variations.
2. Agents often contend with constrained onboard computational resources. This limits the feasibility of large-scale data storage and batch training processes, necessitating approaches that are computationally efficient and tailored to resource limitations.

The above two aspects perfectly fit the design goals of OL.

However, there are differences between OL in robotics and machine learning (Yan et al., 2023). In the field of robotics, “online” emphasizes that robots learn spontaneously and autonomously during operation without human intervention, mainly taking offline learning as a contrast. More specifically, the OL model is used as a robot learns, while the offline learning model will not be updated during use once it is deployed to the robot. OL in robotics can be (small) batch, i.e. learning one or more data at a time. In addition, OL in robotics faces a unique challenge compared to machine learning: the data that enters a learning system is often unannotated.

This dissertation studies OL for 3D object detection in autonomous driving. In addition to the above mentioned two general reasons why OL is suitable for deployment in autonomous driving, its research motivations include:

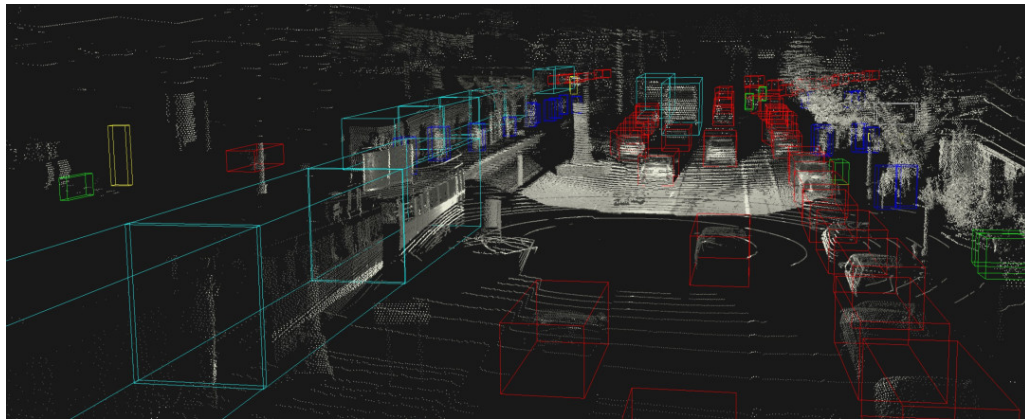


Figure 1.1: Annotated example of a point cloud generated from a 3D LiDAR scan.

3. 3D LiDAR, as the main sensor of autonomous vehicles, its data is difficult to interpret and annotate manually (see Figure 1.1).
4. Long-term deployment of autonomous vehicles poses challenges for knowledge preservation and maintenance in OL systems.

Besides these two additional aspects, the last important motivation driving our research is the long-tail problem, which poses a huge challenge for road participant detection (see Figure 1.2). Although this dissertation does not address the problem head-on, we argue that the approach it studies, i.e. Online Learning, is a powerful way to deal with the long-tail problem. Actually, it is unrealistic to rely only on offline trained models based on manually labeled data in autonomous driving, because these data are inherently difficult to exhaust all objects and corner cases.



Figure 1.2: Long tail examples of road participant detection.

In summary, our research vision actually consists of two aspects. In terms of scientific methods, we hope to explore the usability of OL methods in the field of autonomous driving, starting with 3D object detection (the scope of this dissertation), and eventually expanding to the planning and control of the vehicle in the future. In terms of industrial applications, we hope that our method can reduce the learning and maintenance costs of vehicle loading models, including data collection and annotation, model training and updating, human supervision and fine-tuning, etc., and finally realize the out-of-the-box use of autonomous vehicles.

## 1.2/ CHALLENGES

### 1.2.1/ HOW TO MAKE THE AGENT AUTONOMOUSLY EXTRACT LEARNING SAMPLES FROM SENSORY DATA?

Autonomous vehicles rely on various sensors to perceive changes in the ontology and its external world (Yan et al., 2020c). Measurements from sensors are represented in various data forms, such as images produced by cameras and point clouds produced by 3D LiDARs. These data correspond to the observations of the agent, and are analyzed by the latter to extract useful information. For OL of road participant detection, the agent needs to determine the location and category of the object it wants to learn in each observation, and then extract the data representing the object as a learning sample. This autonomous process is very challenging, especially in the field of autonomous driving, because the urban road environment is usually very complex and highly dynamic, and there is a performance bottleneck only relying on the data analysis of a single observation. How to break through this bottleneck belongs to the scope of this dissertation. Finally, when it comes to 3D LiDAR, since the current mainstream 3D LiDAR can only provide sparse points representing the distance of objects and lacks easy-to-learn features such as color and texture, it is even more challenging to learn online from the data generated by this device compared to other devices such as cameras.

### 1.2.2/ HOW TO MAKE THE AGENT EFFICIENTLY PRESERVE KNOWLEDGE WITHOUT SAVING LEARNING SAMPLES?

Not saving learning data but only saving learned knowledge is a basic setting of OL methods. The learned knowledge is usually presented in the form of models. In the field of autonomous driving, under limited computing resources, the update (training) of the multi-class road participant detection model should be done as fast as possible, and the updated model can be deployed immediately. This involves research on the structure of the model, the distribution of learning samples at a time, the storage and loading of the model, and other aspects.

### 1.2.3/ HOW TO MAKE THE AGENT AVOID CATASTROPHIC FORGETTING IN THE LONG-TERM LEARNING PROCESS?

In OL, or even in any learning method that requires model updating, fresh learning may lead to the problem of degrading the performance of previously learned models, which is known as the catastrophic forgetting problem. The probability of this kind of problem will increase with the variety of learning tasks. In the field of autonomous driving, the long-

term deployment of vehicles will inevitably increase the variety of learning tasks. A typical situation is that the vehicle drives from one scene to another. The problem of catastrophic forgetting has a long history of research in the field of machine learning, and it is also one of the research focuses of the deep learning community in this era (Goodfellow et al., 2013; Kirkpatrick et al., 2017; Li and Hoiem, 2017). However, despite significant advances in specific domains, many approaches to overcoming catastrophic forgetting do not generalize straightforwardly to robotics, or are not feasible at all, due to agents' limited onboard memory and computing resources. Therefore, methods suitable for autonomous vehicles need to be developed, which is the focus of this dissertation. A practical example is shown in Figure 1.3.



Figure 1.3: Road participants of the same category but with very different appearances in East (left) and West (right). After the agent learns in the eastern scene and then updates the previously learned model in the western scene, the object detection performance of the model may decrease after returning to the eastern scene, and vice versa.

### 1.3/ OBJECTIVES

The research objective of this dissertation is to coherently respond to the three challenges identified in the previous section. Scientifically, a proven OL method (Yan et al., 2018) in robotics is being used for autonomous driving for the first time. The core idea of this method is to use one sensor or model to train the other online. Specifically, assume that one detector already has the ability to classify objects, while the other does not and hopes to acquire this ability through OL. To this end, a multi-target tracker is used to loosely correlate detections from different object detectors with confidences that they belong to a certain object class. These confidences are then fused to estimate which object category all detected samples belonging to each object belong to, and the samples are thus annotated with the final labels for OL. The entire process of generating labeled learning samples can be compared to the pretext mechanism in self-supervised learning. The contribution of this dissertation corresponds to the following advancements compared to the previous method (Yan et al., 2018) as well as SOTA:



1. A new spatio-temporal fusion method for multi-modal detection is proposed to efficiently provide high-quality samples for OL. Specifically, spatially, using the calibration of a 3D LiDAR with respect to a camera mounted on a vehicle and the timestamp synchronization of the data produced by the two sensors, the object bounding boxes detected in the point cloud generated by the 3D LiDAR are first projected onto the 2D image produced by the camera, and the Intersection over Union (IoU) is then calculated with the object bounding box detected in the latter to determine whether they are detections of different modalities of the same object. Temporally, detections are correlated in point cloud data using a multi-target tracker tailored for autonomous driving, based on Probabilistic Data Association (PDA), Unscented Kalman Filter (UKF), and Interactive Multiple Models (IMM). This spatio-temporal fusion method enables the agent to quickly generate a substantial number of high-confidence learning samples with limited sample confidence information, thereby improving the efficiency of OL. The source code for the method implementation is publicly available at [https://github.com/epan-utbm/efficient\\_online\\_learning](https://github.com/epan-utbm/efficient_online_learning).
2. After comprehensive investigation, the Online Random Forest (ORF) model was selected, improved, and innovatively integrated into our OL framework, making it possible for the agent to quickly train the model and deploy it immediately under limited computing resources. Specifically, *i*) the requirement that the total amount of data and data distribution be known in the original implementation is removed, and in contrast, support for streaming data processing is added, including online estimation of the optimal training parameters of the Random Forest (RF) model; *ii*) based on *i*), support for few-shot learning and real-time access to the model is added; *iii*) the entire improved implementation is integrated into a system based on the Robot Operating System (ROS) for autonomous driving. Moreover, our exploration of the ORF tree structure shows that using octrees instead of the originally designed binary trees can improve model storage efficiency and access speed. The source code for the method implementation is publicly available at [https://github.com/epan-utbm/octo\\_orf](https://github.com/epan-utbm/octo_orf).
3. For the first time, the catastrophic forgetting problem within the OL framework is explicitly studied, and the corresponding overcoming mechanism is elegantly created without modifying the overall framework. Specifically, an Online Continual Learning (OCL) paradigm named Long-Short-Term Online Learning (LSTOL) is proposed, which combines multiple short-term learning models and a long-term memory controller to enable the agent to learn new knowledge from new data without erasing the previously learned knowledge. The short-term module is based on the concept of ensemble learning and aims to achieve rapid learning iterations, while the long-term module contains a simple yet efficient probabilistic decision-making mechanism combined with four control primitives to achieve effective knowledge maintenance.

nance. A novel feature of the proposed LSTOL is that it avoids forgetting while learning autonomously. In addition, LSTOL makes no assumptions about the model type of short-term learners and the continuity of the data. The source code for the method implementation is publicly available at <https://github.com/epan-utbm/lstol>.

The above contributions have been demonstrated through comprehensive experiments. In particular, two open datasets, KITTI (Geiger et al., 2012) and Waymo (Sun et al., 2020), which are widely recognized in the research community and industry, are used. Three questions are answered from an experimental perspective: 1) What happens to OL in KITTI? 2) What happens to OL from KITTI to Waymo? 3) What happens to OL from Waymo back to KITTI? The strengths and limitations of the proposed methods are analyzed through Online Continual Learning (OCL) and detection of road participants including cars, cyclists, and pedestrians in the two datasets.

## 1.4/ THESIS ORGANIZATION

The rest of the thesis is organized as follows:

- **Chapter 2:** The exploration of existing work related to the dissertation is discussed in terms of upstream and downstream tasks. We provide a comprehensive analysis of the challenges currently faced by online learning and continuous learning as upstream tasks, along with review of existing solutions and state-of-the-art methods. Additionally, we delve into the multi-sensor fusion in object detection and tracking within the context of environment perception in autonomous driving, which serves as a downstream task. Furthermore, we discuss the concept of online continuous learning and its advancements and applications in improving the accuracy and adaptability of object detection.
- **Chapter 3:** In order to facilitate the readers' comprehension of the subsequent content in the dissertation, in the chapter of Preliminary Knowledge, we first elucidate the general framework of the proposed online continuous learning method, which consisting of four closely interconnected components: detection, tracking, learning, and control. Subsequently, we provide an overview of the data used in our experiments, which include KITTI and Waymo datasets. We outline the data pre-processing steps undertaken for training, testing, and validation. Followed by the metrics employed in our evaluation process, including Classification Metrics (e.g. accuracy, precision, recall, and F1-score), Detection Metrics (e.g. Intersection over Union (IoU) and Average Precision (AP)). By presenting this preliminary knowledge, we aim to provide readers with a comprehensive understanding of the foundational

aspects of our research, from the methodological framework to the practicalities of data utilization and the evaluation criteria applied.

- **Chapter 4:** In response to the challenge posed by the interpretability of sparse point clouds generated by 3D LiDAR and the difficulty associated with manual annotation, we presented an efficient online transfer learning method for the 3D detection of road participants in autonomous driving. The framework aims to automatically and efficiently transfer object detection capabilities from 2D monocular camera to 3D LiDAR through a multi-target tracker-based pipeline, enabling knowledge transfer between sensors of different modalities. Through comprehensive experiments, it is proved that our proposed framework can not only enable 3D LiDAR to quickly learn the detection ability of road participants (i.e. cars, pedestrians and cyclists) in the urban environment without ground truth and human intervention, but also leverage multimodal detectors to improve the overall detection of road participants in autonomous vehicles.
- **Chapter 5:** Efficient online model training and access is another key challenge. In this chapter, we establish three general criteria for online learning models: rapid training, immediate deployment, on-the-fly updates and a specific criterion for autonomous driving : explainability. The Online Random Forest (ORF) is innovatively integrated into our online transfer learning system as a specific implementation of the online learning model, which is inherently fast and well-suited for multi-class learning tasks. In addition, we describe in detail the improvement of ORF in two aspects: small batch learning and octree-based model access. Experiments across datasets demonstrate the adaptability of our learning framework to different environments, making it particularly suitable for in-situ deployment. However, the experiment also raised a new challenge: the forgetting problem, which refers to instances where the model's performance from an older dataset (e.g., KITTI) deteriorates after training on a newer dataset (e.g., Waymo).
- **Chapter 6:** To address the accompanied problem of catastrophic forgetting previously learned knowledge in long-term online learning, we propose an ensemble learning framework, named Long-Short-Term Online Learning (LSTOL), which consists of a set of short-term learners and a long-term control mechanism. The former can be any model but needs to be subject to the requirements of online learning, such as fast iteration without saving learning samples. The latter contains a dynamic gate controller that controls whether each existing short-term learner should be updated, kept or removed, or a new short-term learner should be created. The effectiveness of the proposed framework in avoiding forgetting is demonstrated through cross datasets (KITTI and Waymo), on downstream task of 3D classification of road participants, and compared with the above EOTL and baseline of Expert Gate.

## RELATED WORK

### 2.1/ INTRODUCTION

The field of autonomous driving technology has made rapid advancements in the past decade, with one of the key driving forces being the incorporation with machine learning techniques. The latter is driven by data and construct models for various downstream tasks in autonomous driving, including perception, decision-making and control. Typical upstream methods include deep learning, which has garnered popularity due to its outstanding performance in this era. However, a substantial amount of existing efforts has been spent on the construction of deep models and the fine-tuning of hyper-parameters. While these endeavors have undoubtedly contributed to the progress of autonomous driving, few studies have dealt with the autonomous updating of models after they are deployed to vehicles. Furthermore, there has been a glaring lack of emphasis on the research of on Online Learning (OL) within the realm of autonomous driving.

In this chapter, for our proposed online continuous learning, and task-specific in 3D detection of road participants, we give a comprehensive review of the existing related technologies, especially the methods of spanning upstream and downstream tasks, interspersed with our insights and analyses. Our overarching goal is to provide a well-rounded perspective on the current state of research, thereby to further clarify the research positioning of the dissertation.

### 2.2/ UPSTREAM TASKS

Related work on two upstream tasks related to the dissertation is reviewed in this section including OL and Continual Learning (CL).

### 2.2.1/ ONLINE LEARNING

Offline learning entails training a model using complete data, often requiring a substantial amount of manually annotated data to achieve a globally optimal model. Human intervention, such as fine-tuning training parameters and retraining the model, is typically allowed during the offline learning process. Once the model is deployed on the agent, it remains unchanged. In contrast, online machine learning represents a paradigm where the agent can spontaneously and automatically update its knowledge model over time without assuming complete data, relying on annotated data, or involving human intervention (Yan et al., 2023).

As mentioned in Chapter 1, OL emerged in the machine learning community, but its application in robotics thus autonomous driving is very promising because its design concept is very suitable for the needs of the latter. This section first identifies the challenges in the application of OL in the field of robotics, and then gives a review and analysis of existing solutions.

#### 2.2.1.1/ CHALLENGES

**Data acquisition** The challenges involved in data acquisition are twofold: 1) Which sensor data to use? 2) How to interpret the data and extract learning samples? Regarding the first question, a typical autonomous vehicle is equipped with various sensors to sense the vehicle itself and its external environment. The former can use wheel encoders, IMUs, thermometers, etc., while the latter can use devices such as LiDAR, cameras, sonar, etc. According to different downstream tasks, the perceptual data that needs to be learned may be different. For example, the data generated by 3D LiDAR can be learned for object detection, the data generated by camera may be helpful for understanding the semantics of the environment, and it would be better to learn some IMU data for vehicle control. Note that these examples are not exclusive to the use of sensors, i.e. it is certainly possible to learn from multiple (multimodal) sensor data, and in some specific contexts the latter may even be a better solution.

For offline learning, the usual practice follows “data collection - data annotation - model building - model training - model tuning - model deployment”. The worst case scenario is to repeat this workflow from scratch if the model needs to be updated. Two obvious limitations of this approach are the cost of building and maintaining the model and the adaptability of the model. Using an online method like the one studied in this dissertation is an alternative. However, a new question that arises is “when” to use which sensor data? For example, cameras are sensitive to lighting conditions and may not be able to provide usable images in strong light or under poor lighting conditions. Another example is that LiDAR does not perform well in foggy conditions and may not provide valid sen-

sensor readings. Essentially, sensors that measure different physical properties must have their own strengths and weaknesses. Therefore, it would be a good practice to design upstream methods according to the characteristics of downstream tasks.

Regarding the second question, the interpretation of data is a problem that runs from the low level to the high level. The representation of data is an example of the former. For example 3D LiDAR data can be represented both as (grayscale) images and as point clouds. Both representations have advantages and disadvantages. For example, images are easier to process than point clouds but lose data accuracy (due to the lack of information in one dimension). The semantics of data can be an instance of high-level interpretation. For example, if the LiDAR data is represented by a point cloud, the semantics can be extracted based on coordinates or morphology. While the semantics in images usually rely more on color and texture information.

After the sensor data has been properly interpreted, extracting information of interest from it actually consists of two sub-problems: information localization and annotation.

**Self-supervised learning** OL in robotics emphasizes spontaneity and autonomy, *in-situ* and on-the-fly. Many practical challenges surround these ambitious goals. One of the first questions that the robot needs to answer is when to learn and when not to learn. This involves the convergence and stability analysis of OL. The second question is how the robot can learn the models without interrupting its work and use them while learning.

**Fast deployment** Deploying machine learning models to the edge is an open problem, and for some models such as the recently popular Large Language Model (LLM), it is not even possible under the existing technical background. If it is offline learning, such as using Deep Neural Networks (DNN), it may be necessary to optimize the network structure according to the computing resources at the deployment end. While using OL, the computing resources at the deployment end should always be considered throughout the construction of the learning framework, including model selection, to implementation.

### 2.2.1.2/ APPROACHES

The concept of online learning for mobile robots dates back more than two decades, with early work by Thrun Thrun (1994) advocating for lifelong learning capabilities in mobile robots. In recent years, advancements in hardware and algorithms have led to extensive research in online learning for mobile robots Teichman and Thrun (2011); Yan et al. (2017, 2018, 2020a); Majer et al. (2019); Broughton et al. (2020).

Teichman and Thrun Teichman and Thrun (2011) introduced a semi-supervised learning

method based on the Expectation-Maximization (EM) algorithm for the object trajectory classification problem in 3D LiDAR data. They demonstrated that the learning of detection capabilities for dynamic objects can be improved by incorporating information from multi-target tracking systems. Their approach starts with a small set of hand-labeled seed trajectories and a large set of background trajectories pre-collected in areas devoid of pedestrians, cyclists, or cars.

In contrast, our research takes a further step by focusing on *in-situ* learning, which involves training the model directly on the target device or system, utilizing local resources and data. This approach emphasizes capturing the specific characteristics and context of the target environment. In-situ learning is primarily concerned with learning in the absence or with incomplete background knowledge, where background samples are discriminated online as the model learns. On the other hand, online learning is a broader concept that encompasses the dynamic update of the model as new data arrives, regardless of whether the training occurs on the target device or in a centralized system.

From a methodological point of view, compared with offline or batch learning, online learning emphasizes only one or a few samples at a time and the ability to run, train and feedback in real time. From an engineering perspective, online learning must be an incremental process, because its implementation is to continuously update the model through data flow. Therefore, it must have some characteristics of continuous learning, and it will face the challenge of training on new data (maybe new categories are generated, or the categories may not change, but the distribution of old categories has changed).

The connection between online learning and continual learning can be conceptualized as a shared paradigm where both approaches emphasize the dynamic adaptation and improvement of a model over time. The primary principle of this paradigm is to enable models to adapt, learn, and improve continually over their operational lifetime. It acknowledges that the world is dynamic, and models must evolve to remain effective and relevant.

- **Data-Driven Learning:** Both online learning and continual learning rely on data to update and enhance the model's knowledge. Online learning continuously incorporates new data as it arrives, while continual learning operates in a more structured manner but with the overarching goal of adapting to new environment.
- **Resource Efficiency:** Online learning and continual learning aim to make the most of available resources, be it computational power, memory, or data, without requiring retraining from scratch.
- **Adaptation:** Online learning handles real-time adaptation to evolving data, while continual learning accommodates the acquisition of new knowledge and tasks while preserving previous learning.

- **Addressing Catastrophic Forgetting:** In online learning, it's the challenge of adapting to new data without losing knowledge of old data. In continual learning, it's the ability to learn new tasks without significantly degrading performance on previous tasks.
- **Incremental Improvement:** Instead of requiring complete retraining, models learn and improve in smaller, manageable steps in online learning and continual learning. This aligns with the idea of models learning progressively.

Numerous online and incremental algorithms have been published, spanning a diverse array of algorithm categories like linear models, tree ensembles, and neural networks. We'll give a brief overview of these techniques.

Incremental Support Vector Machine (ISVM) is an exact incremental Support Vector Machine (SVM) proposed in (Cauwenberghs and Poggio, 2000). It maintains a set of support vectors alongside a limited set of "candidate vectors". These candidate vectors represent examples that may become support vectors based on future data. The fewer candidate vectors there are, the greater the risk of overlooking potential support vectors. ISVM is a lossless algorithm, meaning it yields the same model as a corresponding batch algorithm when the set of candidate vectors includes all previously encountered data.

In contrast, the Learning Algorithm for Support Vector Machines (LASVM), introduced in reference (Bordes et al., 2005), is an online approximate SVM solver. LASVM operates differently by checking whether the presently processed example qualifies as a support vector and subsequently removing obsolete support vectors. It makes extensive use of sequential direction searches, similar to the Sequential Minimal Optimization (SMO) algorithm (Platt, 1998). Unlike ISVM, LASVM does not maintain a set of candidate vectors but only considers the current example as a possible support vector. This approach offers an approximate solution while significantly reducing training time.

Moving on to an incremental version of the Random Forest algorithm, the Online Random Forest (ORF) (Saffari et al., 2009) is a noteworthy adaptation. In ORF, a predefined number of trees continuously grows by adding splits whenever a sufficient number of samples are collected within a leaf. Unlike traditional Random Forests, which compute locally optimal splits, ORF tests a predefined number of random values based on the Extreme Random Trees scheme (Geurts et al., 2006). The split value that optimizes the Gini index the most is selected. Tree ensembles, such as ORF, are highly regarded for their accuracy, simplicity, parallelization capabilities, and insensitivity to feature scaling. They find significant application, especially in real-time object detection for autonomous driving.

Learn++ (LPPCART) (Polikar et al., 2001) handles incoming samples in predefined chunks. For each chunk, an ensemble of base classifiers is trained and then combined us-



ing weighted majority voting to create an "ensemble of ensembles." This approach is similar to the AdaBoost algorithm (Freund et al., 1999). LPPCART is a model-independent algorithm, allowing various base classifiers like SVM, Classification and Regression Trees (CART) (Loh, 2011), and Multilayer Perceptron (Rumelhart et al., 1985) to be applied.

Incremental Extreme Learning Machine (IELM) takes the batch ELM least-squares solution and reformulates it into a sequential scheme (Liang et al., 2006). It reduces training complexity by randomizing the input weights, similar to the batch version. IELM is compatible with processing data one-by-one or in chunks, which significantly speeds up processing time. However, initializing the output weights requires a sufficient number of examples, at least matching the number of hidden neurons.

Stochastic Gradient Descent (SGD) is an efficient optimization technique used for learning discriminative models by minimizing loss functions like the Hinge or Logistic loss. This approach, especially when combined with linear models, excels in sparse, high-dimensional data scenarios, common in text classification or natural language processing. However, linear models are inadequate when nonlinear class boundaries are needed, frequently the case for low-dimensional data.

Two SVM variations perform similarly, with LASVM capable of processing slightly larger datasets due to its approximate nature. Both face limitations when dealing with large or noisy datasets (Ertekin et al., 2010). Researchers have proposed various extensions for LPPCART and IELM, mostly aimed at addressing non-stationary environments through the introduction of forgetting mechanisms. It is important to note that forgetting can be detrimental and harm overall performance.

In the case of LPPCART, the flexibility of using arbitrary base classifiers and limited knowledge integration across chunks pose challenges. Methods for expediting SGD convergence were presented in (Bottou, 2010), but the results obtained by the SGD algorithm are inevitably influenced by the general benefits and limitations of linear models, such as low model complexity and linear class boundaries.

Table 2.1: Overview of All Relevant Hyperparameters

	Hyperparameter ( <i>independent</i> )	Hyperparameter ( <i>adjustment</i> )
SVMs	Kernel Function	Radial Basis Function ( $\sigma$ ) Regularization
ORF	Split Threshold ( $\theta$ ) Trees (Numbers and Depth)	
LPPCART	Chunk Classifier	Classifier's Parameter Chunk Size
IELM	Activation Function	Hidden node
SGD	Loss Function	Learning Rate

The complexity of model selection varies based on the number and type of hyperparameters. Table 2.1 provides an overview of all relevant hyperparameters, with those adjusting the scale, like learning rates or  $\sigma$  in the Radial Basis Function kernel, being particularly crucial. Independent means that the parameter is not affected by the dataset or task, whereas adjustment means that the parameter needs to be adjusted for the different data distribution. These parameters not only impact accuracy but also strongly influence overall model complexity. Generally, tree-based models like ORF are easy to tune and typically perform well out of the box. In contrast, scale-sensitive models such as ISVM and LASVM require precise, dataset-dependent configuration of multiple parameters to achieve satisfactory results. SGD, on the other hand, focuses on minimizing the Hinge loss function and primarily requires adjusting the learning rate. LPPCART necessitates specifying the number of base classifiers per chunk and the parameters of the base classifier itself.

### 2.2.2/ CONTINUAL LEARNING AND LIFELONG LEARNING

Continual Learning (CL) (also known as incremental learning) refers to continuously learning a large number of tasks without forgetting the knowledge gained from previous tasks, where data from old tasks is not available for training new ones. Lifelong Learning (LL) refers to learning a large number of tasks during the agent's lifetime and transferring knowledge between the learning of different tasks, which is usually task-independent.

Both basically work on the same problem and have been around for a long time, with early attempts dating back to the 90s (Ring, 1994; Thrun, 1994). However, due to the different historical backgrounds, the research emphases of the two are also different. CL emerged in the field of machine learning, focusing on answering how to not forget the previously learned tasks when learning the current one, that is, to avoid catastrophic forgetting. LL begins in the field of robotics, mainly to answer how to use the experience of previous tasks to learn the current task faster and better, such as a closed-loop system. Since our research is positioned at the intersection of machine learning and robotics, and is interested in both of the above two emphases, thus, continual learning, incremental learning, and lifelong learning are investigated as the same topics in this section.

(Ruvolo and Eaton, 2013) gives a architecture of knowledge system in the LL which also could applies to the CL discussed in this dissertation, as shown in Figure 2.1:

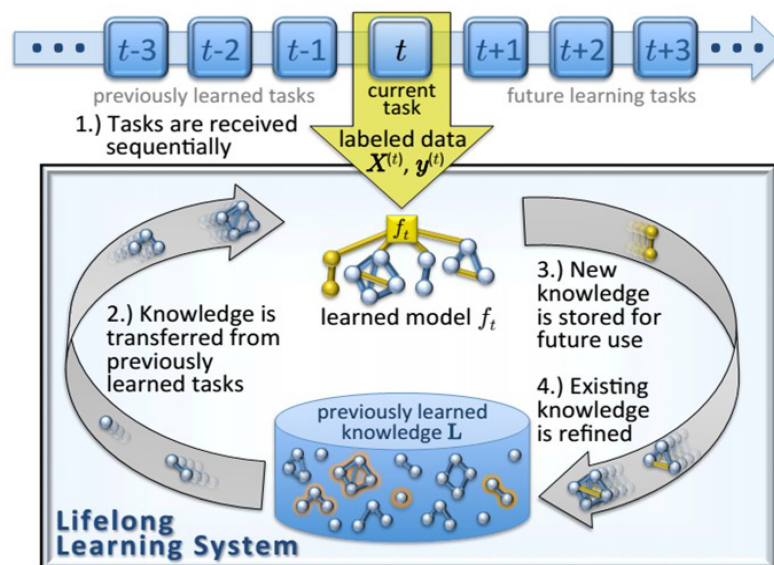


Figure 2.1: Knowledge system in Lifelong Learning in (Ruvolo and Eaton, 2013).

Following this line of thought, in the context of continuous object detection scenarios, we believe that CL should address the following objectives:

- **Limited Resources:** CL models should learn sequentially and utilize bounded memory and computational resources.
- **Backward Transfer:** CL models should be capable of enhancing the performance of previously learned tasks by learning new tasks.
- **Forward Transfer:** CL models should be able to leverage previously acquired knowledge to facilitate the learning of new tasks. This means that the model should not start from scratch every time it encounters a new task but should build on its

existing knowledge to expedite the learning process for new tasks.

- **Rapid Adaptation and Recovery:** CL models should have the capability to quickly adapt to new tasks and guard against catastrophic forgetting.

In short, Continual Learning aims to create models that can retain and transfer knowledge effectively while efficiently expanding their capabilities to handle multiple tasks. This addresses the challenges of learning in a continuous and ever-expanding data environment, enabling long-term adaptability and knowledge retention.

### 2.2.2.1/ CHALLENGES

**Catastrophic Forgetting** Catastrophic forgetting (Robins, 1995) occurs when a model learns different tasks separately in multiple time slices, and as it generalizes to new tasks in later time slices, the performance on old tasks in earlier time slices sharply declines. This phenomenon poses a critical challenge in continuous multi-task learning, as it undermines the ability to retain previously acquired knowledge while learning new tasks. How to prevent catastrophic forgetting is one of the concerns of this dissertation.

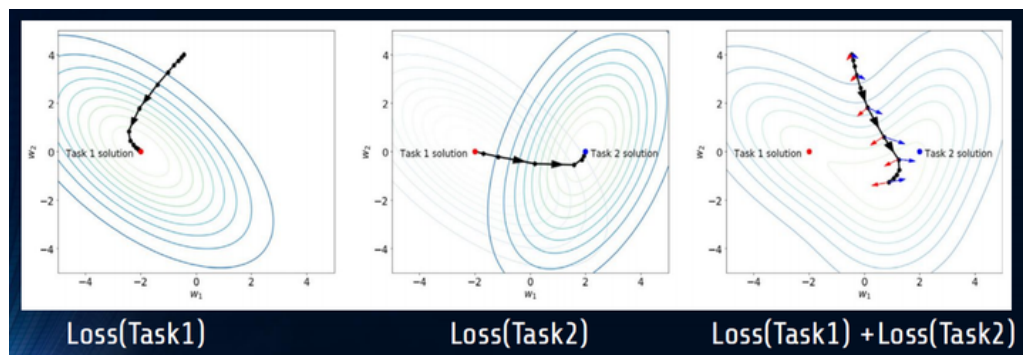


Figure 2.2: Tug-of-War phenomenon illustration (Hadsell et al., 2020): the two figures on the left show the trajectory when optimizing the loss function on a single task, and the right shows the trajectory when optimizing both losses at the same time.

To cope with catastrophic forgetting in the process of CL, it is crucial to analyze the gradient in the multitask optimization problem. In stochastic gradient descent, the dynamic gradient exhibits a Tug-of-War phenomenon (Hadsell et al., 2020), where gradients from different tasks compete for updates, ensuring better results on both tasks simultaneously. However, in a CL condition, tasks appear separately at different time periods, and the training set information from previous time periods is not available at the current time period. This leads to a situation where the model optimizes for the current task, leading to a loss of performance on previously learned tasks, as depicted in the trajectory illustration in Fig.2.2.

**Stability-Plasticity Balance** The stability-plasticity balance in a learning model refers to the interplay between its ability to retain previously learned knowledge (**Stability**) while adapting and optimizing its performance for new tasks (**Plasticity**). This concept is often referred to as the stability-plasticity puzzle.

The biological aspects of human's Continual Learning, as well as the neural network architectures inspired by biological processes of neurons, have motivated the development of long-term learning models. In human brain, neurosynaptic plasticity plays a crucial role in regulating the stability-plasticity balance across various regions. The Fig.2.3 illustrates two forms of neurosynaptic adaptation that play a role in the stability-plasticity balance:

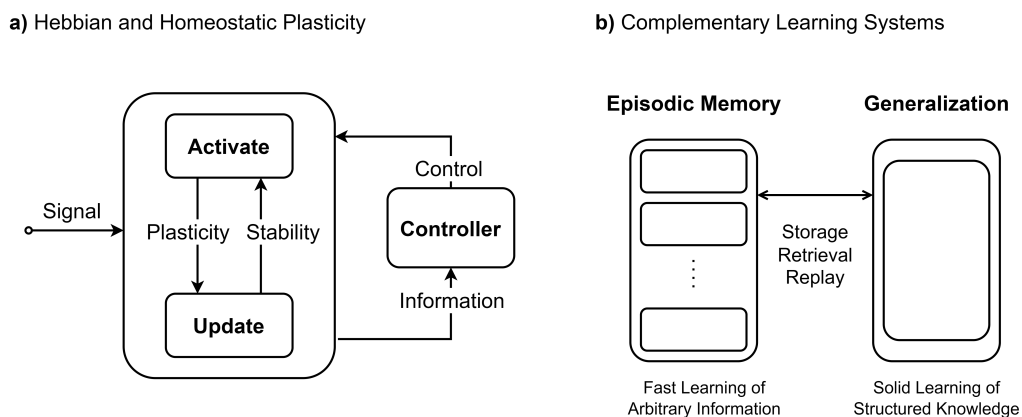


Figure 2.3: The illustration of Hebbian Learning with Homeostatic Plasticity and Complementary Learning Systems

- **a) Hebbian Learning with Homeostatic Plasticity:** This combination allows for the formation of new synaptic connections (Hebbian plasticity) while simultaneously maintaining the stability of existing connections through homeostatic mechanisms.
- **b) Complementary Learning Systems (CLS):** This theory proposes that the brain employs multiple learning systems that complement each other in the learning process. Each system specializes in different aspects of learning, contributing to a balanced and efficient stability-plasticity trade-off.

### 2.2.2.2/ APPROACHES

In the previous analysis of the two main challenges in CL, it becomes evident that CL demands an optimization scheme that considers the influence of previous tasks while performing well on the current task. However, during training, negative interference can lead to catastrophic forgetting, where certain features learned from one task may negatively impact the performance on other tasks. A model must effectively adapt through a single pass over the continuous data stream, during which it may encounter new classes

(referred to as Online Class Incremental (OCI)), or face data nonstationarity, encompassing elements such as new background, blur, noise, changes in illumination, occlusion, and more (known as Online Domain Incremental (ODI)). Based on these findings, We position this dissertation in the Online Domain Incremental (ODI) filed. Then we will discussing solutions related to the challenges focus on three main techniques: regularization-based, architecture-based and replay-based approaches.

- **Regularization-based Approaches:** In scenarios where storing information from previous tasks is not feasible, regularization-based approaches come into play. Cleverly designed regularization losses can be employed to limit the forgetting of old knowledge when learning new task.
- **Architecture-based Approaches:** To avoid forgetting previous tasks, an intuitive approach is to build large enough models and create a subset of the model for each task. This can be achieved by fixing a shared backbone and adding new branches for each new task. This way, the model retains knowledge from previous tasks while learning new ones without interfering with the existing knowledge.
- **Replay-based Approaches:** Replay-based approaches are grounded on the idea of preserving or compressing key data from previous tasks. When learning a new task, these methods mitigate forgetting by replaying the stored samples during training. These samples or pseudo-samples are utilized either for joint training or to constrain the optimization of the loss for the new task, preventing interference with the knowledge from previous tasks.

**Regularization-based Methods** In neural neuroscience theoretical models, the protection and consolidation of learned knowledge from forgetting can be achieved by employing synapse stimulation with different levels of plasticity in a cascading manner. From the perspective of computational systems, this can be accomplished by applying regularization constraints to the model. By using regularization methods to impose constraints during weight updates, it becomes possible to learn new tasks while retaining existing knowledge, thus mitigating the problem of catastrophic forgetting. These regularization methods can be further divided into parameter regularization, distribution regularization, and Bayesian related. Table 2.2 summarizes the representative regularization-based methods in recent years.

Table 2.2: Regularization-based Methods

		References
Regularization-based	Parameter Regularization	EWC (Kirkpatrick et al., 2017)
		SI (Zenke et al., 2017)
		Online-EWC (Schwarz et al., 2018)
	Distribution Regularization	LwF (Li and Hoiem, 2017)
		LwM (Dhar et al., 2019)
	Bayesian Related	Online-LA (Ritter et al., 2018)

### Parameter Regularization Related Methods

(Kirkpatrick et al., 2017) proposed an approach called Elastic Weight Consolidation (EWC), which combines supervised and reinforcement learning. EWC ensures that the model retains knowledge from previous tasks while learning new ones by applying a penalty to the model parameters based on their importance for previous tasks. The schematic diagram of EWC is illustrated in Fig.2.4, where it relates the importance of the parameters to the loss function, then finds a balance parameter that allows effective learning on task B without causing significant loss on task A.

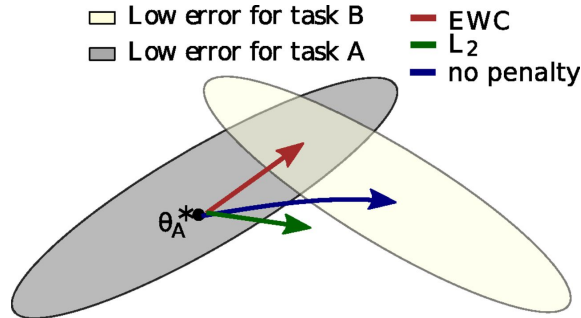


Figure 2.4: The illustration of Elastic Weight Consolidation (EWC)

Specifically, EWC models the parameter  $\theta$  of the task training data set  $D$  through a posterior probability distribution  $p(\theta|D)$ . However, due to the difficulty in estimating such  $p(\theta|D)$ , it is approximated as a Gaussian distribution with a mathematical expectation of  $\theta_A^*$ , and the precision parameter of  $\theta_A^*$  is given by the diagonal element of the Fisher Information Matrix (FIM). Therefore, the loss function of EWC is defined as:

$$L(\theta) = L_B(\theta) + \sum_i \frac{\lambda}{2} F_i (\theta_i - \theta_{A,i}^*)^2 \quad (2.1)$$

where  $L_B(\theta)$  is the loss function for task B,  $\lambda$  represents the balance parameter between new and old tasks,  $i$  is the index for candidate parameters,  $F_i$  represents FIM.

To alleviate the complexity of computing the FIM in EWC, Zenke et al. (2017) proposed a method for online calculation of weight importance known as the Synaptic Intelligence

(SI), which dynamically adjusts the weight of each parameter based on its contribution to the loss during training, where a larger contribution of the parameter  $\lambda_i$  indicates greater importance. The calculation of weight importance as follow:

$$F_k = \frac{\sum_k \Delta L_k}{T_k^2 + \zeta}, \quad (2.2)$$

$$\Delta L_k = \Delta \theta_k \cdot \frac{\partial L}{\partial \theta_k} \quad (2.3)$$

where  $\Delta \theta_k$  is the amount of weight update,  $\frac{\partial L}{\partial \theta_k}$  is the gradient,  $\sum_k \Delta L_k$  indicates the overall loss change,  $T_k$  represents the trajectory of weight  $\theta_k$ , and  $\zeta$  is a small constant to avoid division by zero. Since all the necessary data for computing  $F_k$  is available during the Stochastic Gradient Descent (SGD), there is no additional computational overhead, which effectively reducing the cost.

In addition, (Schwarz et al., 2018) proposed an Online Elastic Weight Consolidation (Online-EWC) model based on the Progress and Compress Framework (P&C). This model is a structurally scalable continual learning method consisting of knowledge repository and active column. The model achieves forward knowledge transfer by iteratively optimizing these two components. They can be seen as layers in a neural network, used for predicting probabilities of the class in supervised learning or generating policies or values in reinforcement learning.

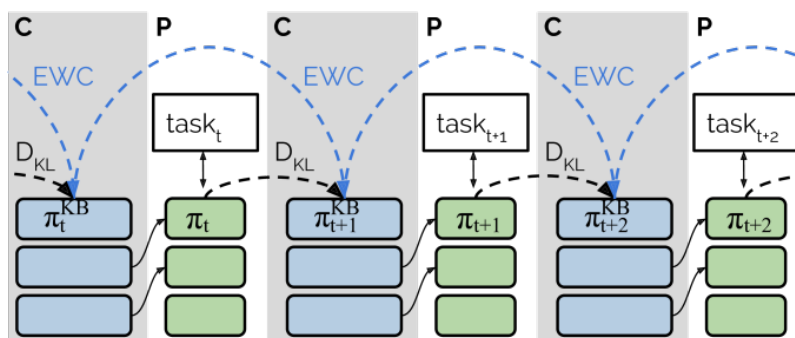


Figure 2.5: The illustration of Online Elastic Weight Consolidation (Online-EWC)

Fig.2.5 illustrates the process of alternating learning between knowledge repository and active column in reinforcement learning. During the "Progress (P)" phase, when learning a new task, the knowledge repository (gray background) is fixed, and the parameters of the active column (grid background) are optimized without applying any constraints or regularization. Notably, a simple knowledge adapter can be used to reuse past learned knowledge in knowledge repository during this process.

In the "Compress (C)" phase, the model performs knowledge distillation, i.e., it transfers newly acquired knowledge forward to the knowledge repository. This phase resembles



the classical EWC, but the Online-EWC model overcomes the linear increase in computational cost with the number of tasks by using an online approximation algorithm to approximate the diagonal FIM.

### Distribution Regularization Related Methods

(Li and Hoiem, 2017) proposed a Learning without Forgetting (LwF) method, which is based on Convolutional Neural Network (CNN). This approach combines Knowledge Distillation (KD) and fine-tuning technique to prevent forgetting of previously learned knowledge.

Given a CNN with shared parameters  $\theta_{share}$  and task-specific parameters  $\theta_{old}$ , the goal is to add task-specific parameters  $\theta_n$  for a new task and only use new data and labels (without using labels from existing tasks) to learn these  $\theta_n$ . The objective is to achieve good predictive performance for both the new and previous tasks. The schematic diagram of the LwF method is illustrated in Fig.2.6.

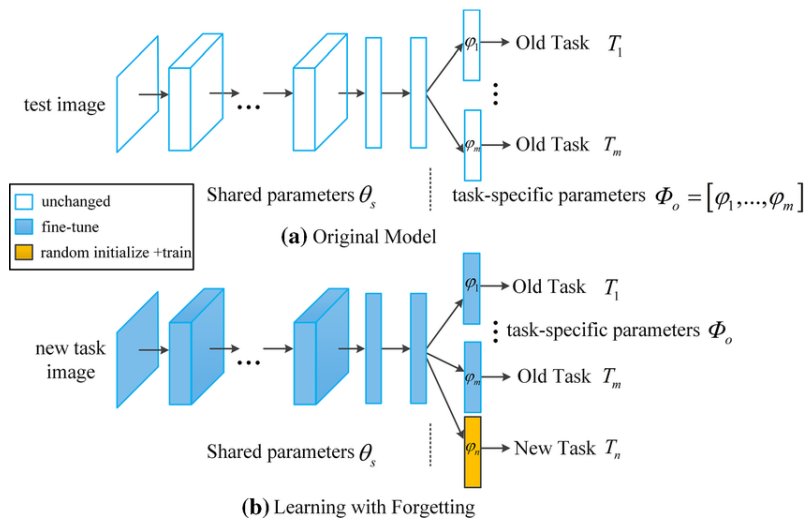


Figure 2.6: The illustration of Learning without Forgetting (LwF)

In addition, (Dhar et al., 2019) proposed a Learning without Memorizing (LwM) method based on attention mechanism mapping. This approach helps the model learn new task incrementally by constraining the difference between teacher and student models. Additionally, the LwM does not require any previous information when learning new task. Different from previous works, the LwM takes into account the gradient flow information between the teacher and student. It utilizes those information to generate attention mechanism mapping, effectively improving the model's classification accuracy.

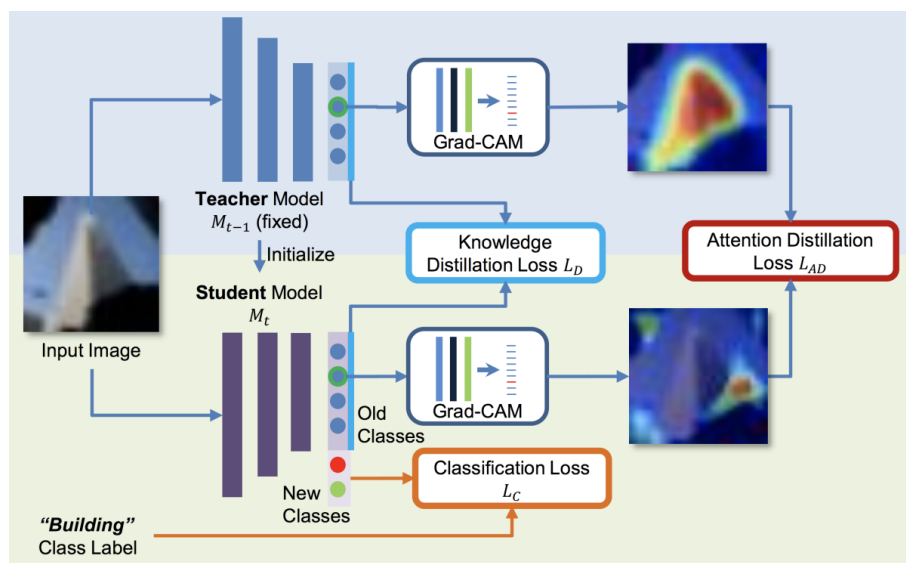


Figure 2.7: The illustration of Learning without Memorizing (LwM)

During the learning process of task  $t$ , the attention-based knowledge preservation term  $L_{AD}$  effectively prevents the student model from deviating too much from the teacher model. To make use of the "hidden knowledge" in the teacher model during student learning, the LwM applies a distillation loss  $L_D$  penalty. The schematic diagram of the LwM is illustrated in Fig.2.7.

### Bayesian Related Methods

(Ritter et al., 2018) proposed an approach called Online Laplace Approximation (Online-LA) using Kronecker factor from a Bayesian perspective to alleviate catastrophic forgetting. This method is based on the Bayesian online learning framework, where they use Gaussian function recursion to approximate the posterior function for each task, resulting in quadratic penalty related to weight changes. Laplace approximation requires computing the Hessian matrix around each mode, which can be computationally expensive. To ensure scalability, Online-LA introduced block-diagonal Kronecker factor approximations of curvature, transforming the complex computation problem. The Maximum a Posteriori (MAP) estimation of the neural network model is given by

$$\theta^* = \arg_{\theta} \max_{\theta} \log p(\theta|D) = \arg_{\theta} \max_{\theta} \log p(D|\theta) + \log p(\theta) \quad (2.4)$$

where  $\log p(D|\theta)$  is the likelihood function of the data, and  $\log p(\theta)$  represents the prior information. In Online-LA, the iteration steps are similar to Bayesian online learning. The Gaussian function is used to recursively approximate the posterior function of each task, and then the corresponding mean and accuracy matrix can be obtained.

**Conclusion** The LwF method only requires using data from new tasks to optimize the model for improved accuracy on these new tasks, while maintaining the neural network’s predictive performance on previous tasks. This method is similar to joint training but does not utilize data and label information from old tasks. Its drawback is that performance highly depends on the relevance of the tasks, and the training time for individual tasks increases linearly with the number of tasks. The EWC method constrains network parameters using the FIM to reduce forgetting of previously learned knowledge, and it does not add any computational burden during training. However, this comes at the cost of computing and storing the LwF values, as well as keeping copies of the previous learning model parameters. Online-LA reduces forgetting and exhibits some scalability.

In conclusion, regularization methods provide a means to mitigate catastrophic forgetting under specific conditions. However, these methods introduce additional loss terms for preserving consolidated knowledge, which can lead to a trade-off issue in performance on old and new tasks when resources are limited. Although distillation methods offer a potential solution for multi-task learning, they still require persistent storage of data for each learning task.

**Architecture-based Methods** Architecture-based long-term learning is a method that involves dynamically adjusting the network or model structure to adapt to a continuously changing environment. This training approach allows for selective training of the model and, when necessary, expands the model to accommodate the learning of new tasks.

### **Copy Weights with re-init (CWR) and CWR+**

(Lomonaco and Maltoni, 2017) proposed a continual learning method called Copy Weights with Reinit (CWR). To avoid interference with the learning of different task weights, CWR sets two sets of weights for the output classification layer: stable weights ( $\theta_{cw}$ ) for long-term memory and temporary weights ( $\theta_{tw}$ ) for rapid learning of the current task.  $\theta_{cw}$  is initialized to 0 before training the first task, while  $\theta_{tw}$  is randomly reinitialized, such as through Gaussian distribution sampling, before each task’s training. In a multi-task continual learning situation, due to the differences between tasks, the weights corresponding to the current task in  $\theta_{tw}$  are copied to  $\theta_{cw}$  after each task’s training. This ensures that  $\theta_{cw}$  serves as a mechanism for long-term memory learning, while  $\theta_{tw}$  acts as a short-term working memory mechanism, facilitating the learning of new task knowledge without forgetting the knowledge learned from previous tasks.

Furthermore, to avoid frequent changes in the weight matrices and bias vectors of shallow connections in the neural network, all shallow layer weights are frozen after completing the training of the first task.

(Maltoni and Lomonaco, 2019) improved the CWR method and proposed CWR+ which

introduce Mean Shift and Zero Initialization techniques. Mean Shift, which automatically compensates each batch weight ( $w_i$ ) by normalizing it with the subtraction of the globally averaged weight learned from all tasks. This eliminates the need for re-normalizing network weights, and experimental results show that this approach achieves better performance compared to other normalization methods.

Additionally, the Zero Initialization, where the weights are initialized to zero instead of the typical Gaussian distribution sampling or Xavier initialization. The experimental results demonstrate that introducing these fine-grained normalization and initialization methods, even as simple as Zero Initialization, can enhance experimental performance to some extent in continual learning scenarios.

### Dynamically Expandable Network

(Yoon et al., 2017) proposed a deep network model called Dynamically Expandable Network (DEN) for continual learning tasks. The DEN model dynamically determines its network capacity while training a series of tasks, allowing it to learn compressed and overlapping knowledge shared among tasks. In continual learning, the most significant feature is that all training examples from previous tasks ( $t - 1$  tasks) are unavailable during the training of the current task  $t$ . Consequently, solving the model parameters  $w^t$  becomes an optimization problem. During the optimization process, the DEN model efficiently retrains training examples in an online manner through selective retraining. When a new task arrives, and the existing features cannot accurately represent it, the network dynamically expands by introducing additional necessary neurons to represent the new task features, as shown in Figure. 2.8. Compared to previous network expansion models, DEN can dynamically expand its network capacity, ensuring it has an appropriate number of neurons to learn different tasks effectively.

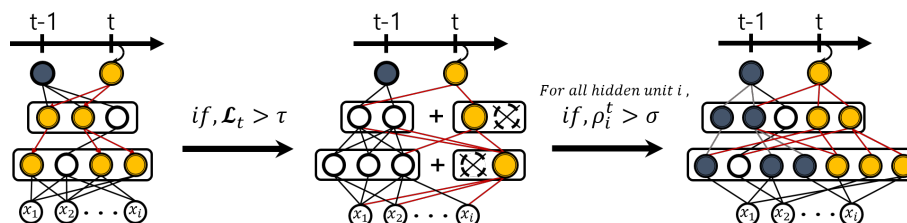


Figure 2.8: The illustration of incremental learning in a Dynamically Expandable Network (DEN): Left: Selective retraining. DEN first identifies neurons that are relevant to the new tasks, and selectively retrains the network parameters associated with them. Center: Dynamic network expansion. If the selective retraining fails to obtain desired loss below set threshold, DEN expand the network capacity in a top-down manner, while eliminating any unnecessary neurons using group-sparsity regularization. Right: Network split/duplication. DEN calculates the drift for each unit to identify units that have drifted too much from their original values during training and duplicate them.

**Expert Gate**

(Aljundi et al., 2017) proposed Expert Gate, aiming to determine which expert network should be used for a new task using an auto-encoder gate. The idea is to select the most relevant old tasks, based on the similarity with the new task, and then perform further training, as shown in Fig.2.9.

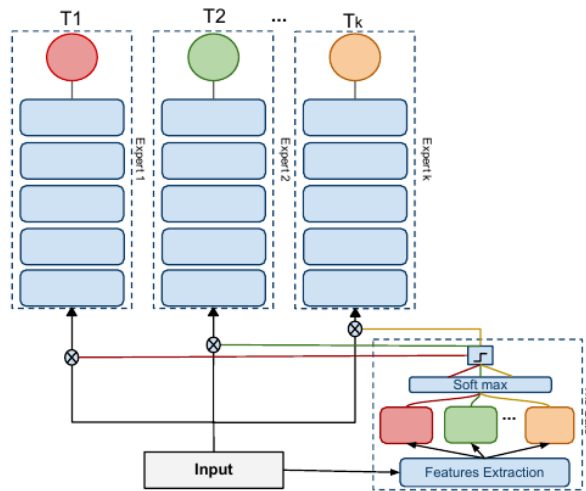


Figure 2.9: The illustration of Expert Gate

Expert Gate is based on the notion that data from a task-related domain should have a lower reconstruction error on an auto-encoder compared to unrelated data. To achieve this, they used different auto-encoders for the new and old tasks and calculated the reconstruction error on each other's data to determine the relatedness between tasks. If the relatedness exceeded a certain threshold, the Learning without Forgetting (LwF) will be applied, otherwise, the fine-tuning strategy will be used .

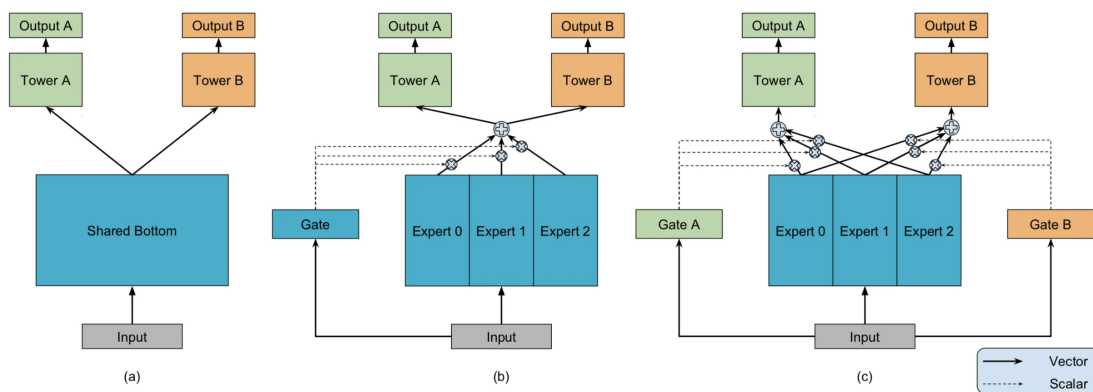


Figure 2.10: The illustration of (a) Shared Bottom, (b) Mixture of Experts (MOE) and (c) Multi-Gate Mixture of Experts (MMOE)

Similar approaches include Mixture of Experts (MOE) (Shazeer et al., 2017) and Multi-Gate Mixture of Experts (MMOE) (Ma et al., 2018). MOE comprises multiple expert networks, each independently contributing to the final output, and a gating network determines the influence of each expert on the target for different tasks. MMOE builds upon the MOE, introducing multiple expert networks for each task, and a gating network specifically learns different combination patterns of expert networks' outputs for each task. The structures of MOE and MMOE are shown as (b) and (c) respectively in Figure.2.10.

**Conclusion** The CWR method achieved the recognition of continuously learning objects, paving the way for subsequent research as a benchmark method. However, a drawback of both the CWR and CWR+ methods is that, after training for each task, some weights are frozen to prevent forgetting of the acquired knowledge. As a result, knowledge cannot be propagated backward, thereby limiting the model's ability to learn new knowledge. While DEN conducts partial retraining of the network for old tasks by explicitly mining inter-task relationships and, when necessary, increases the number of neurons to enhance its capability to interpret new tasks, effectively preventing semantic drift.

These methods effectively mitigate the forgetting problem in continual learning. However, architecture-based methods, as the number of tasks increases, result in an expanding model structure. Therefore, they cannot be applied to large-scale datasets, which represents an important limitation for practical applications of such models.

**Replay-based Methods** The Complementary Learning Systems (CLS) (ref to 2.2.2.1) is a theory that composed of the hippocampus, which emphasizes short-term adaptation, and the neocortex, which maintains long-term memory (Masana et al., 2022). The interaction between the two allows for the rapid acquisition of new knowledge, which is then gradually integrated into long-term memory over time. The CLS theory provides a essential research foundation for modeling memory consolidation and retrieval processes. Inspired by this theory, a series of continual learning models, including episodic memory and generative models, have been proposed.

### Gradient Episodic Memory

(Lopez-Paz and Ranzato, 2017) proposed Gradient Episodic Memory (GEM), which enables forward knowledge transfer from previous tasks and positive transfer of previously learned knowledge to the current task. The key feature of the GEM model is that it stores an episodic memory model  $M_k$  for each task  $k$  to avoid catastrophic forgetting. This model not only minimizes the loss for the current task  $t$  but also uses the loss of the episodic memory model for tasks with  $k < t$  as inequality constraints to prevent an increase in the loss function  $l(f_\theta, D_t)$  while allowing a decrease in the loss function  $l(f_\theta, D_t)$ .

To alleviate the computational burden of the original GEM, (Chaudhry et al., 2018) introduced the Averaged Gradient Episodic Memory (A-GEM). The main feature of the GEM model is to ensure that the loss for each previous task does not increase during each training step. In the A-GEM model, to reduce computational complexity, the aim is to ensure that the average memory loss relative to previous tasks does not increase at each training step.

### Episodic Memory Replay

Episodic Memory Replay (EMR) is an model (Wang et al., 2019) that improves upon the Stochastic Gradient Descent (SGD) by incorporating memory replay. During training on new tasks, EMR randomly samples data from memory and replays it, thereby retaining knowledge from previous tasks in the model. After training each task  $k$ , EMR selects a few training samples to store in the memory  $M$ , denoted as  $M \cap T_{train}^k$ . To address scalability, EMR performs random replay from memory. Specifically, when training task  $k$  with a mini-batch  $D_{train}^k \subset T_{train}^k$ , EMR extracts samples from memory  $M$  to form a second mini-batch  $D_{replay}^k \subset M$ , and then performs gradient updates on both  $D_{train}^k$  and  $D_{replay}^k$ . It is worth noting that EMR can be applied to any stochastic gradient optimization algorithm, such as SGD, AdaDelta, Adagrad, etc.

### Lifelong Generative Modeling

(Ramapuram et al., 2020) proposed the Lifelong Generative Modeling (LGM) which use a Student-Teacher Variational Autoencoder (STVA) based on (Kingma and Welling, 2013) to continuously integrate newly learned distributions into the existing model without the need to retain past data or model structures. This enables the model to learn from the distributions of consecutive tasks. Inspired by Bayesian update rules, they introduced a new Cross-Model Regularizer in the LGM, allowing the student model to effectively utilize information from the teacher model. The LGM employs a dual architecture based on the Student-Teacher model. The teacher's role is to retain distributional memory of previously learned knowledge and transfer it to the student, while the student effectively uses the knowledge acquired from the teacher to learn the distribution of new input data. By jointly optimizing the teacher and student models, this dual system enables learning of new knowledge without forgetting previously acquired knowledge.

**Conclusion** The GEM, compared to the LwF and EWC, yields better performance. However, during training, GEM requires more memory overhead because it involves episodic memory for each task. As the number of learning tasks increases, the training cost sharply rises, and improving performance also imposes a heavier computational burden. The A-GEM, in contrast to the GEM model, ensures that the average episodic memory loss relative to previous tasks does not increase at each training step. This makes it highly memory-efficient, and it does not require storing the additional  $G$  matrix.

The EMR differs from the GEM in that it does not require gradient mapping during unconstrained optimization, such as solving the  $g$ -problem. The time complexity of the EMR is proportional to the number of examples stored for each previous task, whereas the GEM requires computing gradients for all data stored in memory. Therefore, as the number of tasks increases, this computational process grows linearly, making GEM no longer suitable.

The LGM model effectively facilitates knowledge transfer in a teacher-student learning framework. The use of regularization effectively mitigates the problem of interference. However, one issue in the learning process of this model is the inability to access any old data, and all necessary information must be extracted into a single final model to learn.

## 2.3/ DOWNSTREAM TASKS

### 2.3.1/ 3D DETECTION OF ROAD PARTICIPANTS

In recent years, significant progress has been made in 3D object detection using visual sensors thanks to advancements in hardware computing capabilities and the continuous evolution of software and algorithms, particularly the remarkable progress of deep neural networks (Chen et al., 2016; Xu and Chen, 2018; Mousavian et al., 2017). However, despite these impressive achievements, visual sensors have certain inherent limitations that pose challenges to further accurate object detection. One major limitation is their restricted ability to precisely determine object poses, which can be crucial for various applications, such as robotics and autonomous vehicles. Additionally, visual sensors are highly sensitive to changes in lighting conditions, which can negatively impact their performance.

To overcome these limitations and further improve object detection capabilities in some special conditions, researchers have been actively exploring alternative methods that utilize non-visual sensors. An illustrative instance is the work by (Dequaire et al., 2018), who presented a approach utilizing a Recurrent Neural Network (RNN) to grasp the dynamic state of the environment observed through two 3D LiDARs. In an unsupervised manner, they trained a representation model and effectively integrated it into self-driving vehicles to facilitate the detection and tracking of nearby road participants. By exploiting the wealth of spatial information present in 3D LiDAR point clouds, this method significantly bolsters the perceptual abilities of autonomous vehicles.

Similarly, (Chen et al., 2022) presented a method focused on segmenting moving objects in point clouds from 3D LiDAR and automatically generating labels to them. Their approach integrates a multi-target tracker to create trajectories for the moving objects, consequently enhancing the precision of offline coarse detection outcomes. This method



facilitates a better understanding of object movements and interactions, crucial for achieving more reliable and precise object detection results.

Moreover, acknowledging the challenges posed by foggy weather conditions that can impact the effectiveness of camera and LiDAR-based object detection systems, (Majer et al., 2019) proposed an ingenious solution by integrating millimeter-wave radar. They developed a pedestrian detection system using the Support Vector Machine (SVM) and harnessed the radar's capability to penetrate fog, ensuring reliable detections even in adverse weather conditions. This approach significantly enhances the safety and reliability for object detection, particularly in challenging scenarios.

Table 2.3: Advantages and Limitations of Multi-Sensor based Object Detection

Advantages	Increased Reliability	By fusing data from multiple sensors, the system will be more robust and reliable, as it can cross-verify detections and overcome sensor-specific limitations.
	Enhanced Perception	Different sensors provide complementary information, allowing the system to perceive the environment from multiple perspectives, resulting in a more complete and accurate understanding.
	Adaptability	The system can adapt to various environmental conditions, such as changes in lighting, weather, or object appearance, by relying on different sensors that are less affected by specific conditions.
	Redundancy	If one sensor fails or provides erroneous data, the other sensors can compensate for the loss, ensuring the system's functionality and safety.
Limitations	Increased Complexity	Integrating multiple sensors and handling their data fusion requires sophisticated algorithms and calibration, leading to higher implementation and maintenance complexity.
	Higher Cost	Deploying multiple sensors adds to the overall cost of the system, making it potentially more expensive compared to using only one or a few sensors.
	Sensor Synchronization	Ensuring precise data synchronization across sensors can be challenging, requiring careful engineering and calibration efforts.

These research efforts demonstrate the importance of exploring detection methods based

on diverse sensor modalities to address the limitations and challenges associated with visual sensors. By leveraging the strengths of non-visual sensors, such as 3D LiDAR and millimeter-wave radar, which provide unique advantages that complement the strengths of visual sensors, researchers are paving the way for more robust, accurate, and adaptable object detection techniques, thereby advancing the field of autonomous driving and its practical applications.

Certainly, multi-sensors are not a cure-all solution, and they come with their own set of issues and challenges. The application of multi-sensors should be chosen selectively based on the specific requirements and conditions. Tab.2.3 provides a summary of the advantages and limitations associated with using multiple sensors.

By harnessing the unique advantages of multiple sensors and integrating their outputs, autonomous systems can achieve a more comprehensive and accurate perception of their surroundings. Generally, a multi-sensor-based autonomous system designed for object detection and tracking typically involves following several steps, as illustrated in Fig.2.11.

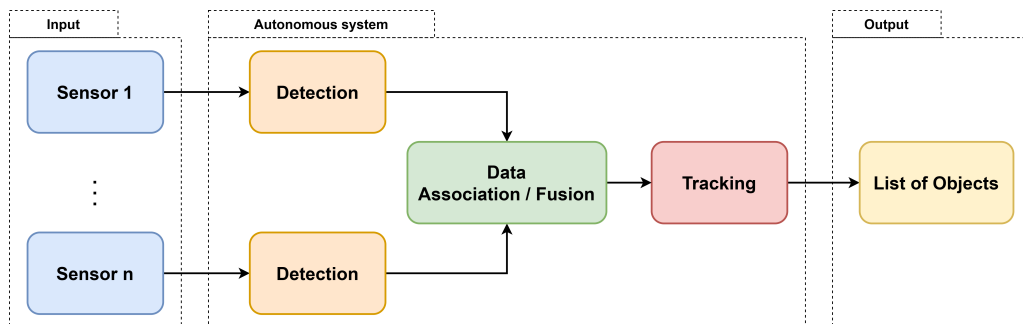


Figure 2.11: Schematic diagram of a typical multi-sensor based autonomous system for detection and tracking

- **Sensor Calibration:** The calibration involves aligning the coordinate systems and intrinsic/extrinsic parameters of each sensor to ensure that the collected data can be effectively combined.
- **Object Detection:** Each sensor modality contributes its specialized information for object detection. Cameras are well-suited for recognizing object appearances, while LiDAR excels at providing precise 3D location information.
- **Data Association/Fusion:** The collection and synchronization of data from different sensors involve matching detections based on time, location, speed, and other attributes, associating the detections of the same object across different sensors. In practice, this data fusion can be performed at various levels, including sensor data fusion, feature-level fusion, or decision-level fusion.
- **Object Tracking:** Once data association is established, the system tracks the

detected objects over time to maintain their identity and trajectory using multiple-target tracking algorithms, such as Kalman filters or particle filters.

The exploration of multimodal sensor perception and data fusion is essential to leverage the unique strengths of different sensors and improve the overall perception capabilities of autonomous driving systems. (Yan et al., 2020c). In this dissertation, our primary focus is on two types of sensor data: images from cameras and point clouds from 3D LiDAR. 3D LiDAR is used because it provides highly accurate spatial information, direct depth perception, robustness in challenging conditions, and enhanced object discrimination, making it a valuable sensor for precise and reliable object detection and tracking in various applications, particularly in autonomous driving and robotics. Through a detailed comparison presented in Table 2.4, our aim is to elucidate the advantages and limitations of these sensor modalities in their applications within the field of autonomous driving.

Table 2.4: Comparison in Advantages and Limitations of Cameras and 3D LiDAR:

	CAMERA	3D-LIDAR
Advantages	Rich visual information	Precise location information
	Low-cost and widely available	Unaffected by lighting conditions
	Established vision-based methods	Direct depth information
Disadvantages	Sensitive to lighting conditions	Higher cost
	Limited depth perception	Sparse appearance information

Over the past few years, there has been a significant increase in detection based on cameras and 3D LiDAR, and these methods have demonstrated improved accuracy.

As an example, (González et al., 2016) conducted a study where they explored the combination of RGB images from a monocular camera and point clouds from 3D LiDAR for object detection. To achieve this, they trained separate Random Forest (RF) for each view and later merged all the RFs into an ensemble. By integrating the information from both images and point clouds, their approach significantly improved the accuracy and dependability of detecting cars, pedestrians and cyclists. This was possible due to the complementary nature of the data sources used in their fusion-based detection framework.

Similarly, (Qi et al., 2018) proposed a method called Frustum PointNets that also employs the combination of an RGB camera and a 3D LiDAR sensor. However, their approach differs in its strategy. They begin by generating 2D object region proposals from RGB

images and subsequently extend each 2D region into a 3D frustum to extract the corresponding point cloud data from the 3D LiDAR. Then, they utilize PointNets (Qi et al., 2017a) to estimate 3D bounding boxes based on the points within the frustum. The schematic diagram of the Frustum PointNets is illustrated in Fig.2.12.

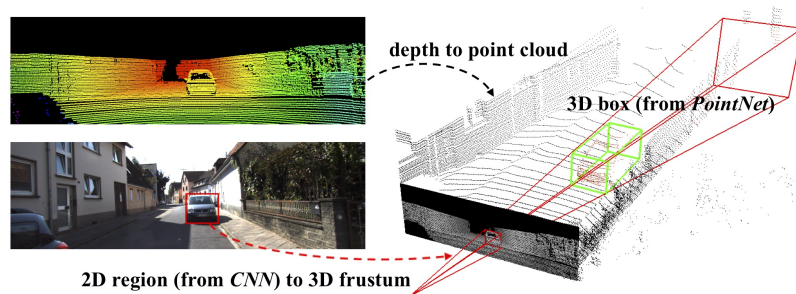


Figure 2.12: The Illustration of Frustum PointNets. 2D object region proposals firstly generated in the RGB image using a CNN. Each 2D region is then extruded to a 3D viewing frustum. Finally, the PointNet predicts a (oriented and amodal) 3D bounding box for the object from the points in the frustum.

This methodology shares some similarities with our knowledge transfer approach, particularly in transferring knowledge from the RGB camera to the 3D LiDAR. However, the primary distinction lies in the scope of information considered. While Frustum PointNets concentrates on instant spatial information (specifically, object detection), our approach incorporates both spatial and temporal information (i.e., trajectories).

## 2.4/ DISCUSSION

### Online Transfer Learning for Road Participant Detection

Multi-Sensor based object detection and tracking is a comprehensive approach that leverages data from multiple types of sensors to enhance the accuracy, robustness, and reliability of perceiving the surrounding environment in autonomous driving and robotics applications. By fusing information from different sensors, such as cameras, 3D LiDAR, radar, and ultrasonic sensors, etc., the system can overcome individual sensor limitations and gain a comprehensive understanding of the environment.

Our investigation of existing work has revealed a significant gap in the field of efficient and autonomous learning for road participant detection, especially concerning dynamic objects, a critical aspect in the context of 3D LiDAR-based autonomous vehicles. To address this need, we propose leveraging transfer learning Pan and Yang (2009), a technique that can enhance the learning process of a target predictive function by leveraging knowledge from a source domain and learning task during the training phase. The primary objective of transfer learning is to adapt the model to the target domain and task,

facilitating better generalization and improved performance in the target setting.

In our context, we focus on Online Transfer Learning, a particularly valuable approach when labeled data is scarce in the target domain or when the target domain experiences changes over time. The choice between traditional transfer learning and online transfer learning depends on factors such as the availability of labeled data, the dynamic nature of the target domain, and the specific requirements of the learning task.

An intuitive idea that we explore in our research is to use visual sensors to train non-visual sensors. The non-visual sensors learn in a self-supervised manner using unlabeled data, supplemented with annotations from the visual sensors. This knowledge transfer enables the non-visual sensors to improve their object detection capabilities efficiently.

Formally, our proposed learning framework can be described as an Online Transfer Learning paradigm. By continuously updating the model based on new data and knowledge from the visual sensors, the framework adapts to changing environmental conditions and improves detection performance over time.

Integrating Online Random Forest into our learning framework holds particular significance due to its inherent advantages. Random Forest is renowned for its fast training speed, suitability for multi-class learning, and ease of interpretability, aligning with the expectations of both the academic and industrial communities for components in autonomous vehicles. The inclusion of ORF complements our online transfer learning approach, ensuring robust and efficient object detection in the context of autonomous driving.

### **Gating-based Long-Short-Term Online Learning**

The ensemble learning concept of Expert Gate provides us an open-ended approach for handling continual learning tasks has inspired our work. Our proposed framework, called as Long-Short-Term Online Learning (LSTOL), is built upon this inspiration, which involves constructing multiple short-term learning modules as "experts" and a long-term control module to manage and utilize these experts effectively. However, there is a key difference between our approach and the ensemble learning methods mentioned earlier. In our framework, the expert networks are developed based on the objectives of multi-task learning, aiming to leverage knowledge across related tasks.

The distinctive aspect of our approach lies in the context of long-term learning, where both data and the environment may be unknown and subject to continual changes. Unlike traditional multi-task learning, which assumes access to all tasks during training, our framework accounts for the challenges of Continual Learning, where tasks are encountered sequentially, and access to past data may be limited. By combining short-term learning modules as experts and a long-term control module to manage knowledge retention and adaptation, our approach seeks to address the complexities of continual learning in dy-

dynamic and evolving environments. The potential applications of this framework include scenarios where models need to learn and adapt over extended periods while dealing with diverse and unpredictable tasks and data distributions.





## CONTRIBUTION





# 3

## PRELIMINARY KNOWLEDGE

### 3.1/ THE METHODOLOGY

A common assumption in machine learning methods includes that the distribution of training and test data remains consistent. However, real-world situations frequently challenge this assumption due to environmental shifts, the introduction of novel objects, or alterations in data distribution. Online Learning (OL) is considered an effective way to avoid this assumption, since it allows continuous refinement and adjustment of one or more models as new data becomes available. The general form of the objective function for OL can be defined as:

$$Regret = \sum_{t=1}^T f_t(w_t) - \sum_{t=1}^T f_t(w_*) \quad (3.1)$$

means that every time  $t$  samples are learned,  $w$  is updated to obtain  $w_t$ , where  $w$  is the set of model parameters. OL should pursue the smallest cumulative regret (i.e. regret minimization), which is similar to the greedy algorithm. This is clearly different from batch learning, where the objective function is:

$$Loss = \sum_{t=1}^T f_t(w_T) - \sum_{t=1}^T f_t(w_*) \quad (3.2)$$

Compared to batch learning, OL has two key assumptions that do not hold: 1) All data is available; 2) Ground truth is available.

A fundamental challenge in using OL in robotics is how to obtain learning data (Yan et al., 2023). This challenge is particularly prominent in the field of autonomous driving: in addition to complex road situations, the difficulty of interpreting lidar data is also an important reason. Second, how to efficiently train, save, and maintain models after the data is acquired constitutes a second challenge. Finally, since OL can be a long-term iterative process, how to learn from new data without forgetting useful knowledge already learned poses a third challenge. In essence, when a new task is learned, the model tends to overwrite the previously learned  $w$  (see Figure 3.1), leading to a decline in performance for the tasks learned in the past.

Motivated by the above three research challenges, with 3D detection of road participants in autonomous driving as a downstream task, this dissertation proposes an Online Continual Learning (OCL) framework with three key features:

- **High-quality learning sample generation without human intervention:** Similar to self-supervised learning, the samples that the agent needs to learn are completely autonomously generated, including extracting the Region of Interest (ROI) from the raw sensor data, and then labeling the region with high confidence. Details on this feature are given in Chapter 4.

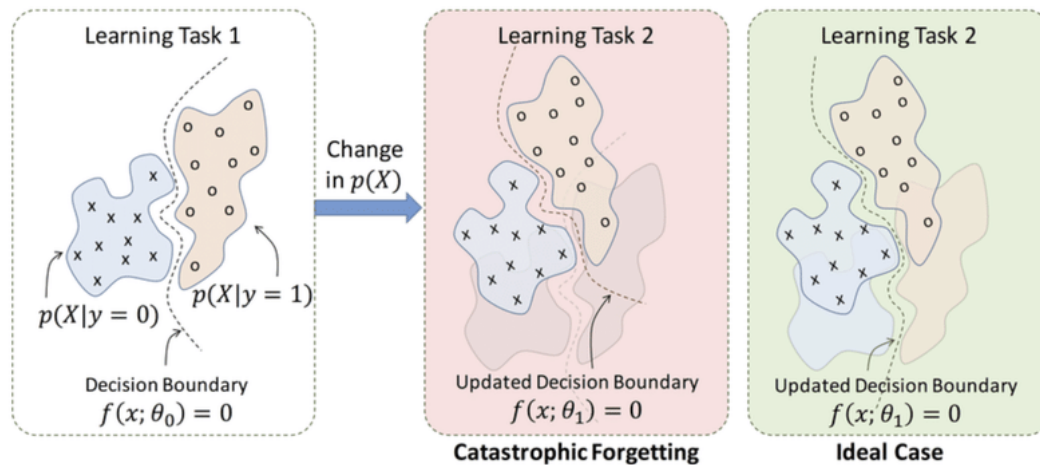


Figure 3.1: Schematic diagram of catastrophic forgetting, which occurs when the system significantly loses performance on previously learned tasks as it learns new information (Kolouri et al., 2019).

- Fast iterative training of multi-class models and efficient model access:** Based on the guiding idea of “learning quickly and applying the learned model immediately” (Yan et al., 2023), an online random forest method is integrated, which is inherently fast and suitable for multi-class learning. Furthermore, how is efficient model access achieved? Details on this feature are given in Chapter 5.
- Efficient prevention of catastrophic forgetting:** Each fresh object category or tracking scenario is regarded as a distinct learning task (based on the idea of multi-task learning), models can be trained on different segments of the data stream to form a classifier ensemble, and individual short-term models are maintained by a set of control strategies. Details on this feature are given in Chapter 6.

The framework is comprised of four closely interconnected components: detection, tracking, learning, and control, as shown in Figure 3.2.

- Detection:** The detection component is responsible for identifying and localizing objects in the input sequential data streams. It utilizes object detection algorithms to detect objects of interest and provides this detection information to the tracking component.
- Tracking:** The tracking component takes the detection information from the detection module and generates trajectories for the objects across consecutive frames. Various tracking algorithms, like Kalman filters or particle filters, can be employed for this purpose.

- **Learning:** The learning component analyzes the trajectories generated by the tracking module. It leverages this trajectory information and incorporates its own learning state information to identify patterns, features, or representations relevant to the tracked objects' behaviors and appearances. Chapters 4 and 6 cover work involving one and multiple learners, respectively.
- **Control:** The control component plays a crucial role in guiding the learning process and preventing catastrophic forgetting. It receives the trajectory information and learning state from the learning module. Based on this information, the control component guides the learning process to generate high-quality samples that can be used for detection training.

The updated detection information from the learning process enhances the tracking module's performance by providing more accurate object detections. This improved tracking, in turn, generates better trajectories for learning. Thus, a closed-loop structure of detection-tracking-learning-control is formed, with each component benefiting from the others' outputs. The four components work together to facilitate continuous improvement and online adaptation of a vehicle's road participant detection performance.

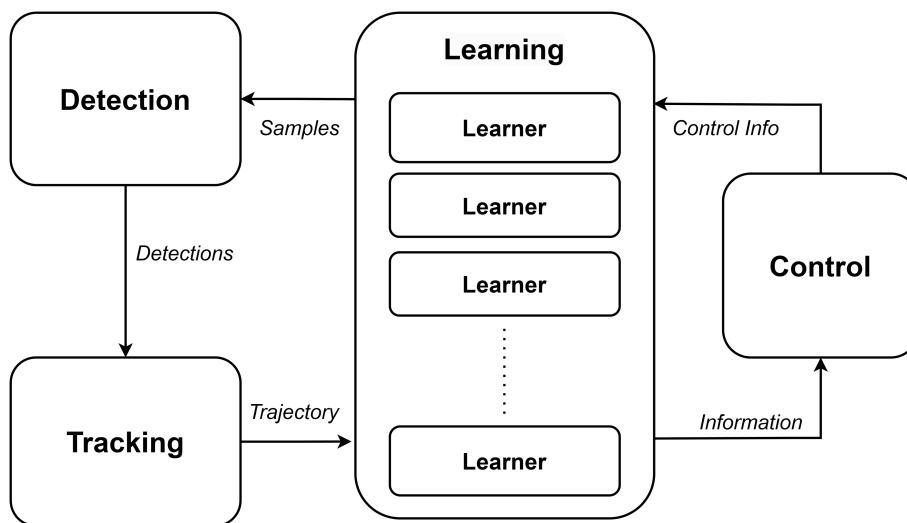


Figure 3.2: Componentized overall structure of the proposed OCL framework.

As a conclusion, OCL is a cutting-edge approach, not only for the field of autonomous driving, but also in a broader sense (Mai et al., 2022). This method elegantly integrates OL and Continual Learning (CL), so that the model can be updated in time with the data, knowledge rather than data can be saved anytime and anywhere, and catastrophic forgetting can be avoided in the long-term learning process.

## 3.2/ DATA USED FOR EXPERIMENTS

### 3.2.1/ OPEN DATASETS

Open datasets have been widely recognized as the key to advancing research in related fields and achieving fruitful results. They typically have standardized formats and evaluation metrics, creating a common ground for researchers to test their methods against existing state-of-the-art ones. Two popular open datasets including KITTI (Geiger et al., 2012) and Waymo (Sun et al., 2020) are used in our research to extensively evaluate the efficacy of the proposed framework. These two datasets encompass real-world complexities and challenges, making them crucial for understanding the strengths and limitations of our methods.

The KITTI dataset (see Figure 3.3 (Geiger et al., 2013)) stands as a pioneering work in the realm of autonomous driving datasets, providing essential exteroceptive data in both image and point cloud formats, which are tightly synchronized. The dataset also offers a comprehensive benchmarking suite, streamlining the comparison of various methods in the field. However, it is important to acknowledge that KITTI does have some inherent limitations. While it serves as an excellent resource for computer vision-oriented approaches to autonomous driving, it might not fully capture the intricate nuances of the autonomous driving task itself. For instance, the dataset handles sensor data with varying acquisition frequencies by employing one-to-one frame synchronization, and the labeling process involves projecting information from images onto the point cloud, which may not entirely reflect the point cloud's true geometric characteristics (Yang et al., 2022). Moreover, KITTI's scenes tend to be conventional and low in dynamic complexity, featuring favorable weather and lighting conditions. These scenes do not fully represent the challenges faced in real-world driving. Additionally, the object detection and tracking benchmarks in the dataset are discontinuous in time, and lack essential global position information about the vehicle itself. Despite these limitations, KITTI remains an indispensable resource for the community. KITTI data were collected in Karlsruhe, Germany.

The Waymo dataset (see Figure 3.4 ), introduced in 2019, represents a significant advancement in autonomous driving data resources. Notably, it offers a scene-based organization, which means it provides continuous annotated data for each scene, enabling a more comprehensive understanding of the driving context. A distinguishing feature of this dataset is the inclusion of real-time global positioning information of the vehicle, enhancing the spatial awareness of algorithms during evaluation. Additionally, the Waymo dataset employs a many-to-one synchronization approach, effectively handling data from various sensors and ensuring a coherent and unified dataset. One of the standout qualities of the Waymo dataset is its diverse and extensive collection of driving environments. By incorporating various weather conditions and capturing data at different times of the

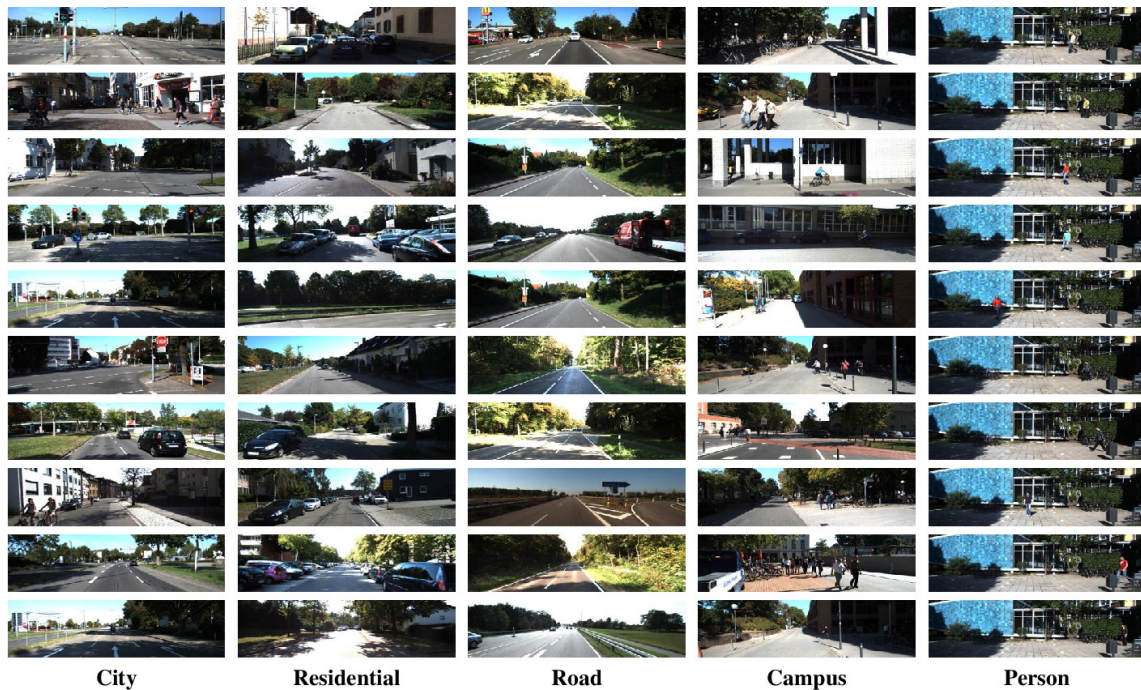


Figure 3.3: Example images from the left color camera in the KITTI dataset.

day, it presents an excellent opportunity for researchers to evaluate algorithms across a wide range of scenarios. This diversity allows for a more robust assessment of the proposed framework’s performance, especially in handling challenging real-world driving conditions. Waymo data were collected in California, USA.

As a summary, the main motivations for using the KITTI dataset in our research include: 1) its wide usage and academic recognition, and 2) the mature benchmark suite it provides to facilitate quantification of results and comparison with peers. The main motivations for using the Waymo dataset include: 1) its data is more in line with the needs of the field of autonomous driving, 2) its diverse driving scenarios, 3) its different scenario from KITTI makes it possible to evaluate the cross-environmental learning and adaptability of the proposed OCL framework, and 4) learning across datasets can help us evaluate the proposed framework’s ability to prevent catastrophic forgetting.

### 3.2.2/ DATA SELECTION

Data used for experiments should be relevant to a specific task. To this end, for the KITTI and Waymo datasets, unannotated continuous raw sensor data are used as input to the OCL framework, while each dataset’s respective evaluation set is used to evaluate the learning performance of OCL. Specifically, in KITTI, scenes containing cars, cyclists, and pedestrians in urban road environments are prioritized, while highways and monotonous scenes dominated by stationary vehicles are excluded, which are usually used for visual



Figure 3.4: Example images from the color camera in the Waymo dataset.

odometry tasks.

In Waymo, we adopt a distinct strategy. Here, we systematically sample 15 clips from the expansive training set, organizing them based on scene type. This meticulous sampling approach serves as the foundation for the online updating of classifiers that were initially learned using the KITTI dataset. Considering the format disparity between the Waymo and KITTI datasets, an essential preprocessing step involves the transformation of the original Waymo dataset into a format compatible with the KITTI dataset. This adaptation streamlines the data loading, model training, and evaluation processes, paving the way for effective knowledge transfer across disparate datasets.

The depth-rich 3D point clouds, forming an integral part of the KITTI detection benchmark, play a pivotal role in the assessment of LiDAR-based detectors.

### 3.3/ EVALUATION METRICS

Evaluation metrics play a crucial role in the development and advancement of research, which provide a systematic and quantitative way to measure the performance of algorithms, models, and systems. Their establishment enables us to objectively compare different approaches and determine which ones are more effective in solving specific tasks. The following subsections detail the evaluation metrics used in this dissertation.

#### 3.3.1/ CLASSIFICATION PERFORMANCE EVALUATION

The confusion matrix is invaluable in understanding the classification performance of the classifier at a micro level, as it provides a more intuitive representation of the differences between the actual labels and the predicted labels for different classes. In the confusion



matrix, each row represents the true labels of the data instances in a specific class, while each column corresponds to the predicted labels made by the classifier for that class.

	Predicted Positive	Predicted Negative	
Actual Positive	<b>TP</b> <i>True Positive</i>	<b>FN</b> <i>False Negative</i>	<b>Sensitivity</b> $\frac{TP}{(TP + FN)}$
Actual Negative	<b>FP</b> <i>False Positive</i>	<b>TN</b> <i>True Negative</i>	<b>Specificity</b> $\frac{TN}{(TN + FP)}$
	<b>Precision</b> $\frac{TP}{(TP + FP)}$	<b>Negative Predictive Value</b> $\frac{TN}{(TN + FN)}$	<b>Accuracy</b> $\frac{TP + TN}{(TP + TN + FP + FN)}$

Figure 3.5: Examples of a standard binary classification confusion matrix

Figure 3.5) presents a typical confusion matrix example for a binary classification problem, as well as the calculation of the other classification performance matrix including Accuracy, Precision, Sensitivity (Recall), Specificity and Negative Predictive Value. The four important basic concepts used in the calculation process are:

- True Positive (TP): A positive example is correctly predicted as positive.
- False Positive (FP): A negative example is incorrectly predicted as positive.
- False Negative (FN): A positive example is incorrectly predicted as negative.
- True Negative (TN): A negative example is correctly predicted as negative.

For the multi-classification problem involved in this dissertation, we transform it into multiple two-classification problems for discussion, and introduced new evaluation indicators. That is, for a certain class in a multi-classification problem, focus on the following three indicators:

- True Positive (TP): Samples belonging to this class are correctly predicted for this class.
- False Positive (FP): Samples that do not belong to this class are incorrectly predicted to this class.
- False Negative (FN): Samples belonging to this class are incorrectly predicted to other class.

F1-score is an evaluation metric used in statistics to measure the accuracy of a binary classification classifier. It is used to measure the accuracy of imbalanced data that takes into account both the precision and recall of the classification model, which can be seen

as a weighted average of model precision and recall. (Its maximum value is 1 and the minimum value is 0.)

In multi-classification problem, there are two calculation methods for calculating the F1-score of a classifier, namely Micro-F1-score (macro-average) and Macro-F1-score (macro-average).

Micro-F1-score is calculated as follows:

$$MicroF1 = 2 \cdot \frac{P_{micro} \cdot R_{micro}}{P_{micro} + R_{micro}} \quad (3.3)$$

while Macro-F1-score is calculated as follows:

$$MacroF1 = 2 \cdot \frac{P_{macro} \cdot R_{macro}}{P_{macro} + R_{macro}} \quad (3.4)$$

where precision and recall are calculated separately for each class from macro and micro perspectives:

$$P_{micro} = \frac{\sum_{i=1}^n TP_i}{\sum_{i=1}^n TP_i + \sum_{i=1}^n FP_i} \quad (3.5)$$

$$R_{micro} = \frac{\sum_{i=1}^n TP_i}{\sum_{i=1}^n TP_i + \sum_{i=1}^n FN_i}$$

and

$$P_{macro} = \frac{1}{n} \sum_{i=1}^n P_i \quad (3.6)$$

$$R_{macro} = \frac{1}{n} \sum_{i=1}^n R_i$$

where  $P$  is precision,  $R$  is recall,  $n$  is number of classes, TP is true positive, FP is false positive, and FN is false negative. As can be seen, Micro-F1-score (macro-average) first calculates the total Precision and Recall of all class, and then calculates the F1-score, while Macro-F1-score (macro-average) first calculates the average of Precision and Recall from each class, and then calculates F1-score.

If the Average Accuracy (ACC) of the model is defined as the proportion of correctly classified samples among all samples, then ACC is mathematically consistent with micro-F1-score. For the convenience of readers' understanding, we will use ACC to replace micro-F1-score in the figures. Similarly, macro-F1-score is actually mathematically the average representation of macro-F1 for each class, so it can be represented by Macro Average (MaA).

ACC is an overall measure consideration of correctly classified samples, but will be affected by data imbalance (for example, there are far more samples of vehicles than cyclists). And Macro Average (MaA) indicate the consistency in classification performance of different classes, which calculates the precision and recall rate of each class separately, so the classes will be treated equally and is less affected by data imbalance, but

it is easily affected by classes with high recognition (high recall or high precision). Using both ACC and MaA for evaluation allows us to gain a comprehensive understanding of our model's classification performance. By combining these metrics, we can ensure that our classification approach is robust and reliable across various class distributions and scenarios.

### 3.3.2/ DETECTION PERFORMANCE EVALUATION

Object detection is an outcome that arises from the collaborative interplay of classification and localization tasks.

In real-life scenarios, road participants encompass several object categories, such as vehicles, pedestrians, and cyclists. Hence, it becomes essential to assign a confidence score to each detected object as an evaluation of classification performance. Typically, different confidence ranges (i.e. classification thresholds) are assigned for each category. Lowering the classification threshold results in decreased precision but increased recall, as objects with lower confidence scores might be redetermined as valid detections, vice versa.

Furthermore, for a more robust assessment of the detector's localization capabilities, it's crucial to determine if the true labels and detection results match. In practice, a metric called Intersection over Union (IoU) is employed to determine the match between two bounding boxes. This metric measures the overlap between each bounding box returned by the detector and all the true bounding boxes, calculating both the intersection and union areas of the two rectangles, as shown in Figure. 3.6.

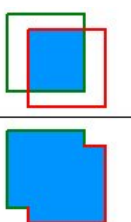
$$IOU = \frac{\text{area of overlap}}{\text{area of union}} = \frac{\text{area of intersection}}{\text{area of union}}$$


Figure 3.6: Intersection over Union (IoU) described in (Padilla et al., 2020).

If the IoU value for a given label detection exceeds the specified IoU threshold, it's classified as a TP. Conversely, when the IoU is below the threshold, the detection is considered a FP. If a true label is entirely undetected, it's categorized as a FN. Changing the IoU threshold also influences precision and recall since a lower threshold leads to many detections matching the labels, resulting in a higher number of TPs. In most benchmarks, like KITTI and Waymo, the minimum IoU required for vehicles is set to 70%.

**Average Precision (AP)** is used to measure the model's detection performance for a

single category. A higher AP indicates better detection accuracy for that category. It is described as follows:

#### a. Precision-Recall Curve

As mentioned earlier, precision and recall vary depending on the classification thresholds. By plotting precision on the vertical axis and recall on the horizontal axis, various points on the PR curve can be obtained by selecting different confidence ranges. AP is calculated by finding the area under the Precision-Recall curve. The definition is as follows:

$$AP = \int_0^1 \text{precision}(\text{recall}) d\text{recall} \quad (3.7)$$

#### b. Interpolation

When calculating the area under a PR curve, interpolation is often performed to smooth the curve. A common interpolation method is to select the maximum recall rate at a fixed precision value, which means that for different precision values, select the precision value corresponding to the maximum recall rate on the curve. The definition is as follows:

$$AP_R = \frac{1}{|R|} \sum_{r \in R} \rho_{interp}(r) \quad (3.8)$$

where in the KITTI setting, 11 equally spaced recall levels are applied, i.e.,  $R_{11} = \{0, 0.1, 0.2, \dots, 1\}$ .

#### c. Mean Average Precision (mAP)

mAP refers to the average of AP values for different categories. For the KITTI dataset, AP values are typically calculated for different difficulty levels (Easy, Moderate, and Hard) and then averaged as mAP. Similarly, mAP can be calculated by averaging the AP values for different classes like vehicles, pedestrians, and cyclists. The idea is to calculate independent AP values for each category and then calculate the overall mean.

For the purpose of meticulous evaluation, KITTI employs a categorization of detection difficulties that factors in diverse attributes such as the bounding box height, occlusion level, and truncation level. These attributes collectively shape the complexity of the detection task. KITTI defines three distinct levels of detection difficulty: "easy," "moderate," and "hard." The categorization is structured in a way that, for instance, in the "easy" level, the objects possess larger bounding boxes with minimal occlusion and truncation. This nuanced stratification enables a comprehensive evaluation that accounts for varying degrees of challenge encountered in real-world detection scenarios.

### 3.4/ CONCLUSION

In conclusion, the methodology discussed in this section introduces the concept of Online Continual Learning as a cutting-edge approach to address the challenges posed by changing data distributions in machine learning, especially in the context of object detection in autonomous driving. This dissertation proposes a framework that integrates Online Learning and Continual Learning to allow models to be updated in real-time with new data, store knowledge rather than data, and avoid catastrophic forgetting. This approach is essential in scenarios where the distribution of training and test data is not consistent due to environmental shifts, introduction of novel objects, or alterations in data distribution.

The research identifies and addresses three main challenges in using Online Continual Learning in robotics, particularly in autonomous driving: the generation of high-quality learning data without human intervention, efficient training and maintenance of models after data acquisition, and preventing catastrophic forgetting during long-term iterative learning. The framework introduced in this section comprises four interconnected components: detection, tracking, learning, and control, working together in a closed-loop structure. The detection component identifies and localizes objects, the tracking component generates object trajectories, the learning component analyzes trajectories and incorporates knowledge, and the control component guides the learning process and prevents catastrophic forgetting. This collaborative approach enables continuous improvement and online adaptation of a vehicle's road participant detection performance.

In addition to the methodology, the section discusses the datasets used for experiments, emphasizing the importance of open datasets like KITTI and Waymo. These datasets provide real-world complexities and challenges, making them essential for evaluating the proposed framework's efficacy. Finally, the section details the evaluation metrics employed to assess the performance of the framework, including classification and detection performance metrics.

# EFFICIENT SAMPLE GENERATION FOR ONLINE LEARNING

## 4.1/ INTRODUCTION

Object detection plays a critical role in environment perception for autonomous driving systems, especially the detection of road participants such as cars, pedestrians and cyclists. It also serves as a foundational component for other essential tasks, including obstacle avoidance, trajectory prediction, and social behavior analysis. Over the recent years, remarkable progress has been made in visual-based object detection methods, which are capable of providing accurate 2D bounding boxes in images, thus determining the location and category of various objects in the driving scene (Ren et al., 2015; Redmon and Farhadi, 2018). However, despite the advancements in 2D object detection, this 2D information alone falls short of meeting the requirements for real-world autonomous driving. To achieve more accurate environment perception and enable more sophisticated vehicle path planning and control strategies, it becomes essential to incorporate 3D information. The latter includes critical details such as the length, width, height, and deflection angle of objects, enabling autonomous vehicles to better handle complex scenarios, such as estimating potential collision risks by more accurately analyzing the spatial relationship between objects. An intuitive example is shown in Figure 4.1.



Figure 4.1: Schematic diagram of 2D detection in images and corresponding 3D detection in point clouds.

In the past decade, 3D lidar, as a provider of 3D information, has become increasingly popular in the field of autonomous driving (Qian et al., 2022). This sensor is capable of providing high-accuracy, long-distance and wide-angle distance measurements, and is superior to visual sensors in terms of robustness to different lighting conditions. However, unlike visual sensors, which can capture color and texture features, lidar-generated data represents the environment as a sparse set of points, making object detection based on the latter challenging. This challenge becomes even more significant in dynamic environments (Yan et al., 2020c; Krajník et al., 2017) (such as daily driving on urban roads) or when traversing different environments (Yan et al., 2020b; Sun et al., 2018) (such as driving from one country to another). Object detection models trained offline usually suffer from the aforementioned challenges, and tedious and costly model maintenance, including but not limited to tuning or retraining, is unavoidable. On the other hand, manually annotating 3D lidar data is also a tedious, costly, and error-prone task, especially when

multiple variations in object pose, shape, and size need to be accurately segmented and labeled.

Our idea is to use the OL method to solve two problems at once: 1) lidar data is difficult to interpret and annotate; and 2) the model trained offline lacks the ability to adapt to the environment. Specifically, the vehicle continuously updates its model by learning new samples on-site and on-the-fly, without human intervention, to achieve efficient 3D detection of various road participants including pedestrians, cyclists, and cars. Figure 4.2 shows a conceptual diagram of our proposed approach. The core idea is that, based on the fact that cameras are a standard feature of autonomous vehicles and the availability of various off-the-shelf image-based object detectors, under the framework of OL, let the camera act as a trainer to help the learning and maintenance of the 3D lidar-based object detector. Scientifically, this learning paradigm is defined as Online Transfer Learning (OTL), since it involves a pipeline of knowledge transfer, that is, a multi-target tracker, to enable the propagation of object detection capabilities from cameras to 3D lidars.

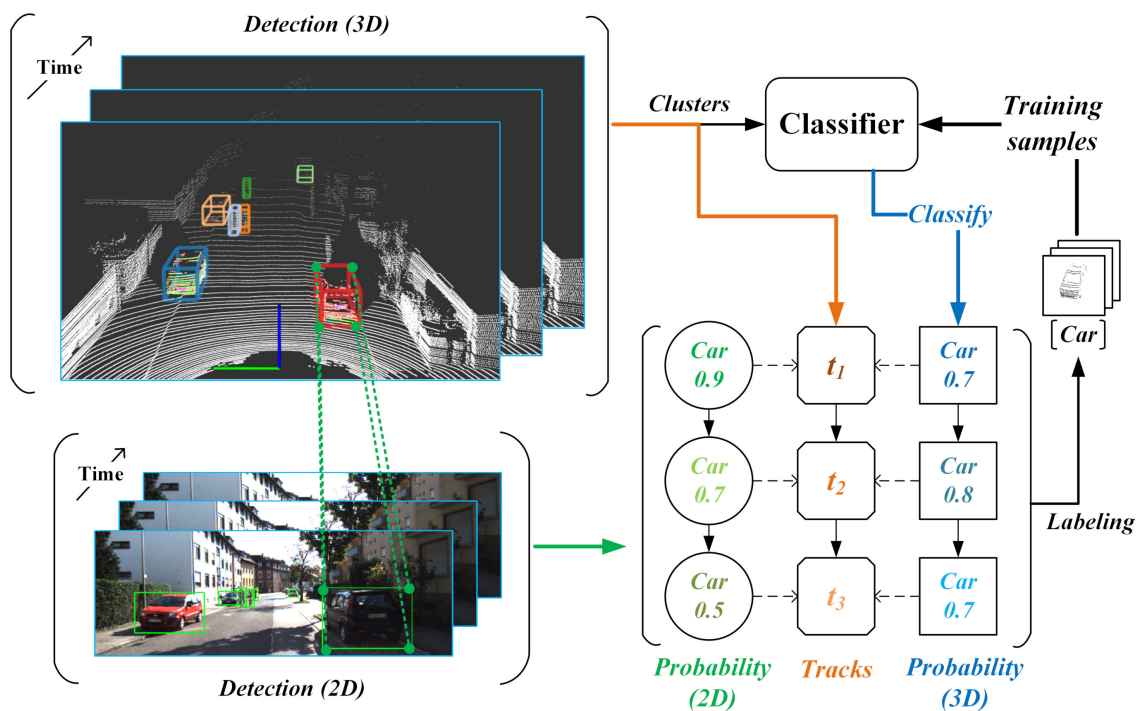


Figure 4.2: Concept diagram of transfer learning from images to point clouds. Initially, different detections of the same object, including the output of an off-the-shelf 2D image-based detector (shown in green) and the output of an online-learned 3D point cloud-based detector (shown in blue show), are matched according to their spatial relationships. Subsequently, detections in different point cloud frames are temporally correlated using a multi-target tracker. Probabilistic estimation is then performed on the trajectories generated by the tracker (shown in orange) to generate the labeled samples required for OL. Finally, the classifier is trained using ORF. The latter is deployed immediately to form a closed loop system.



In engineering, the point cloud generated by 3D lidar is first segmented into disjoint 3D clusters, where each cluster ideally corresponds to an object in the environment. The online learned detector is then used to estimate the probability of the object class to which these clusters belong. Simultaneously, an image-based object detector is applied to the 2D images captured by the camera, providing the 2D bounding boxes of the detected objects and their corresponding class confidences. A key step in the overall learning system is to correlate different detections. Novelty, a spatio-temporal information correlation method is developed, in which spatial correlation (i.e. 2D and 3D) of detections from different sensors at the same moment is first implemented using a calibration matrix between sensors. At this point, the class confidences of the 2D detections has been **transferred** to the 3D detections. A multi-target tracker is then used to temporally correlate 3D clusters with different timestamps representing the same object. Finally, the class confidences of all samples associated with the same target are probabilistically fused to determine their labels as learning samples. The Online Random Forest (ORF) (Saffari et al., 2009) is integrated into our system as a learning model. The entire OL process is iterative, the learning model is learned and used at the same time, and its performance can converge and remain stable in a short period of time.

## 4.2/ PROBLEM FORMULATION

In response to the challenges raised in Section 4.1, or more globally, the challenge described in Section 1.2.1, our research mainly addresses two problems: 1) From a macro perspective, how to achieve knowledge transfer in OL? 2) From a micro perspective, how to realize a pipeline for knowledge transfer? Below we formalize these two problems.

### 4.2.1/ ONLINE TRANSFER LEARNING

The first problem can be formalized as an OTL problem. Specifically, according to (Pan and Yang, 2009), transfer learning can be formalized as follows.

Given a source domain  $\mathcal{D}_S$  and its associated learning task  $\mathcal{T}_S$ , and a target domain  $\mathcal{D}_T$  along with its learning task  $\mathcal{T}_T$ , the objective of transfer learning is to enhance the learning process of the target predictive function  $f_T(\cdot)$  within the target domain  $\mathcal{D}_T$  by leveraging the knowledge gained from the source domain  $\mathcal{D}_S$  and its learning task  $\mathcal{T}_S$ , where  $\mathcal{D}_S \neq \mathcal{D}_T$  or  $\mathcal{T}_S \neq \mathcal{T}_T$ .

We want this enhancement to be done online, denoted as  $OL(f_T(\cdot))$ . More concretely, let a trained detector  $d_i$  be able to detect an object  $x_i$  and assign it a label  $l_i$ , denoted as  $\mathcal{W} = (x_i, l_i, d_i)$  where  $l_i \in L$ , which corresponds to the knowledge gained from  $\mathcal{D}_S$  and  $\mathcal{T}_S$ .

Suppose  $x_i$  is also detected by another detector  $d_j$  but the input data of  $d_j$  is different from that of  $d_i$  (i.e.,  $\mathcal{D}_S \neq \mathcal{D}_T$ ), and  $d_j$  cannot provide a label for  $x_i$  (e.g., due to the output is subject to a confidence threshold). This case is denoted as  $U = (x_i, l_j, d_j)$  where  $l_j = \emptyset$ , corresponding to the learning task  $\mathcal{T}_T$  in the target domain  $\mathcal{D}_T$ . The idea of OTL is to use  $x_i$  as a link to let  $d_i$  help  $d_j$  learn, so that the latter can provide a label  $l_i \in L$  for  $x_i$  as well, denoted as:

$$\sum_{n=1}^N f : U \xrightarrow{W} L \quad (4.1)$$

#### 4.2.2/ INFORMATION FUSION

The second problem can be formalized as an Information Fusion (IF) problem. Specifically, since our formalization in the previous section assumes that there is a common object  $x_i$  in both the source domain and the target domain, the realization of the knowledge transfer pipeline can be further formalized as an IF problem generated by different detectors acting on  $x_i$ . Let  $Y$  denote the detections generated by a finite set of detectors  $D = \{d_1, d_2, \dots, d_n\}$  that continuously process various sensory data over time. Let  $X$  be all objects that can be detected by the detector in the environment, then each detection  $y$  is defined as follows:

$$y_t^d = (x_i, l, b, d, t) \in Y \quad (4.2)$$

where  $x_i \in X_d$  indicates the object detected at a specific time  $t$  by a particular detector  $d$ ,  $l$  represents the classification posterior assigned by the detector  $d$  to the detected object  $x_i$ , and  $b$  denotes the coordinates of the detected object  $x_i$  with respect to the sensor's reference frame. The fusion of information extracted from different sensory data can thus be formalized as:

$$\bigcap_{p=1, p \neq q}^n \{x_i\} = X_{d_p} \cap X_{d_q} \quad (4.3)$$

which represents a intersection of multiple detection of the same object  $x_i$  by different detectors.

### 4.3/ GENERAL FRAMEWORK

The general framework of OL via knowledge transfer is shown in Figure 4.3. It mainly follows (Yan et al., 2018) and comprises four distinct modules, each serving a specific purpose, which are explained below.

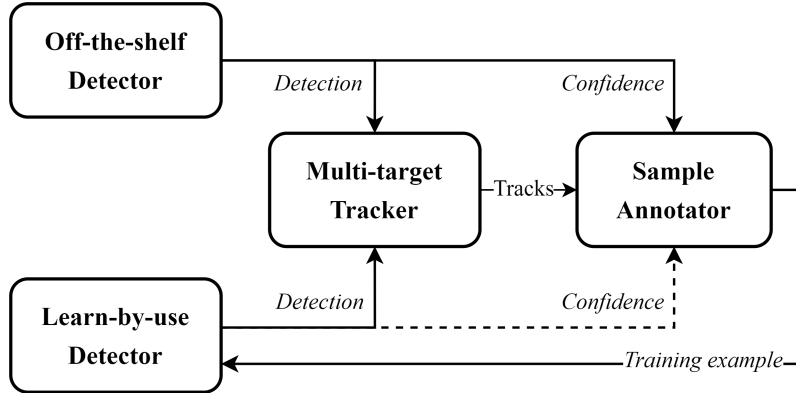


Figure 4.3: Block diagram of the OTL framework. The dashed line indicates that the learn-by-use detector is able to provide increasingly higher detection confidence as it learns online.

**Off-the-shelf Detector** ( $D_{off}$ ) represents a pre-trained detector. In theory, the better the performance of the detector (for example, it can often provide high-confidence detection samples), the faster the performance of the detector that needs to be learned online will be improved. In the research of this dissertation,  $D_{off}$  is mainly embodied as a visual sensor-based object detector that has been pre-trained on annotated data.

**Learn-by-use Detector** ( $D_{learn}$ ) represents a detector that needs to be learned online without human intervention and is used while learning. Since the samples learned by  $D_{learn}$  contain the confidence information provided by  $D_{off}$ , it can also be regarded as the learning of  $D_{learn}$  being supervised by  $D_{off}$ . By design,  $D_{learn}$  should improve its performance through continuous learning.  $D_{learn}$  is embodied as a non-visual sensor-based detector in the research of this dissertation.

**Multi-target Tracker** ( $T_{multi}$ ) plays a key role in the framework, which correlates detections from various detectors in space and time, thus establishing a pipeline for knowledge transfer from  $D_{off}$  to  $D_{learn}$ . The working principle of  $T_{multi}$  can be formalized as:

$$Track_{x_i} = \begin{bmatrix} y_{t-m}^{d_1} & \cdots & y_t^{d_1} & \cdots & y_{t+n}^{d_1} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ y_{t-m}^{d_i} & \cdots & y_t^{d_i} & \cdots & y_{t+n}^{d_i} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ y_{t-m}^{d_k} & \cdots & y_t^{d_k} & \cdots & y_{t+n}^{d_k} \end{bmatrix}_{x_i} \quad (4.4)$$

which means that  $T_{multi}$  can generate a trajectory  $Track$  about object  $x_i$ , and  $Track_{x_i}$  is composed of spatio-temporal association of detections from different detectors  $\{d_k \mid k \in \mathbb{N}^+\}$  at different times  $t_i$ . Intuitively, the rows of the matrix in Equation 4.4 represent the temporal association of the detections, while the columns represent their spatial association. Furthermore, the detection association in any dimension can be loose, meaning

that if a detector  $d_k$  works at a specific time  $t_i$  but does not detect an object  $x_i$ , then “no detection” is also considered a legal output from  $d_k$ .

**Sample Annotator ( $A_{sample}$ )** estimates in real time the class of each object track  $Track_{x_i}$  – actually all the object samples on this track – and then labels them and feeds them to  $D_{learn}$  for OL. The working principle of  $A_{sample}$  is visually depicted in Figure 4.4. Specifically, given an object track containing time-series detection samples from different detectors, the object can be detected by one or more detectors at any time  $t$  and the confidence of the category to which the object belongs is given at the same time. The labels of all the samples on the track are determined by fusing the confidences of all the samples. The samples provided by  $D_{learn}$  are sent back to itself for OL. The samples provided by  $D_{off}$  will be given final labels.

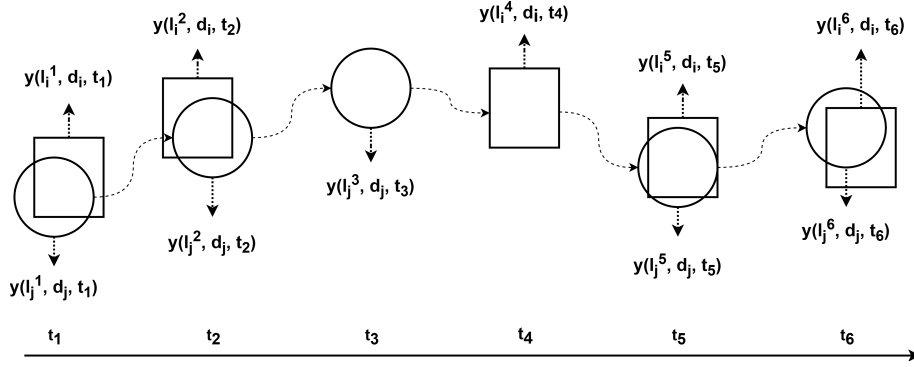


Figure 4.4: Schematic diagram of working principle of the Sample Annotator. The squares represent one detector and the circles represent the other.

It can be seen that  $A_{sample}$  is actually a response to Equation 4.1, while  $D_{off}$ ,  $D_{learn}$  and  $T_{multi}$  can be regarded as parameters on which its implementation depends. Among them,  $D_{off}$  constructs the labeled set  $\mathcal{W}_w$ ,  $D_{learn}$  builds the unlabeled set  $\mathcal{U}_u$ , and  $T_{multi}$  implements the association between the two.

Finally, looking at the proposed framework as a whole, the OL process continues in an iterative manner. Starting from  $\theta_0$ , in each iteration  $k$ , the unlabeled set  $U_u$  is labeled with the help of the model  $\theta_{k-1}$  trained in the previous iteration, and then the model  $\theta_k$  is updated. This process is formalized as:

$$l_u^k = f(u|\theta_{k-1}), u \in U_u \quad (4.5)$$

The process iterates until convergence criteria (discussed in Section 5.5.2) or other stopping criteria (such as the maximum number of iterations) are met.

## 4.4/ IMPLEMENTATION

In this section, we present a concrete implementation of the proposed general framework, in order to respond to the two issues mentioned previously: 1) 3D lidar data in autonomous driving is difficult to interpret and annotate, and 2) models trained and maintained offline lack adaptability to changing environments. The implementation is fully based on the Robot Operating System (ROS) with a very high degree of modularity using C++ and Python mixed coding, which is made publicly available<sup>1</sup> to the community for further exploration and utilization. The detailed system block diagram of the proposed implementation is depicted in Figure 4.5, where each block is described in detail below.

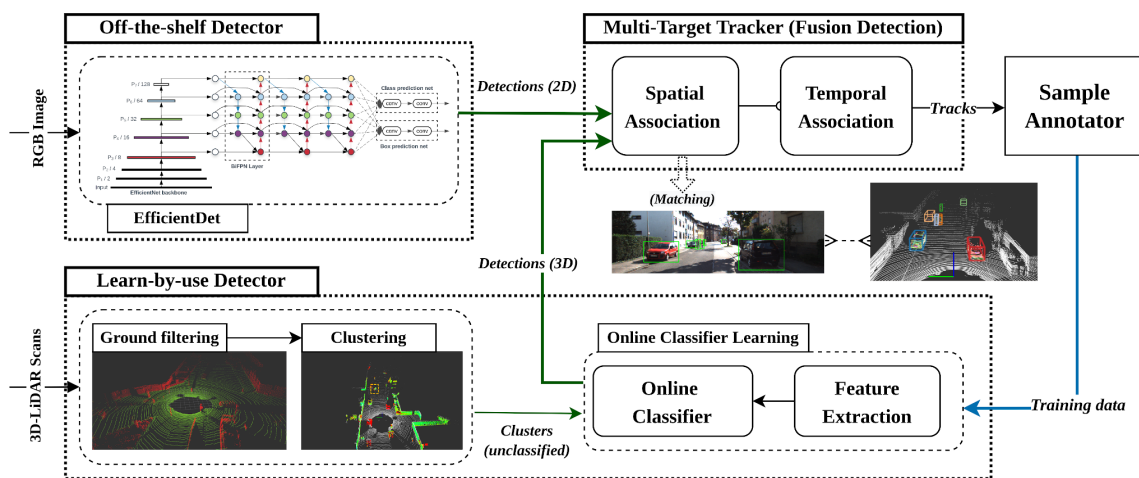


Figure 4.5: Detailed system block diagram of efficient OTL for object detection based on multimodal perception. The RGB image is fed to an off-the-shelf visual detector to get the bounding box of the object in the 2D image and its probability of belonging to an object. At the same time, the point cloud generated by 3D lidar is ground-filtered, segmented into different clusters and then sent to the multitarget tracker. In the latter spatial association, the 3D bounding boxes of the clusters are projected into synchronized 2D image frames to match the visual detection. The clusters are then input to a temporal association module to generate object trajectories. Next, the sample annotator estimates the probability that the trajectory belongs to a certain class, so as to determine the labels of all the samples on the trajectory, which is used for online learning of the learn-by-use detector. The green line represents detection, and the blue line represents learning, which together constitute the closed-loop structure of online learning.

### 4.4.1/ OFF-THE-SHELF DETECTOR

The off-the-shelf image detector is based on a state-of-the-art deep learning model to ensure its high-performance object detection. This detector focuses on detecting cars, cyclists, and pedestrians within 2D images and provides corresponding 2D bounding boxes along with detection confidences, including class labels and prediction scores. To

<sup>1</sup>[https://github.com/epan-utbm/efficient\\_online\\_learning](https://github.com/epan-utbm/efficient_online_learning)

strike a balance between the accuracy and the real-time performance of the detector, we conducted extensive testing using well-established methods: YOLOv3 (Redmon and Farhadi, 2018), Faster R-CNN (Ren et al., 2015), and EfficientDet (Tan et al., 2020). Each method was evaluated on the widely recognized KITTI dataset (Geiger et al., 2012), which contains benchmarks for object detection performance in autonomous driving scenarios.

Table 4.1 shows the evaluation results including mean average precision (mAP) of three difficulty levels for each class and average processing time per frame for object detection across three classes including cars, pedestrians and cyclists. To ensure transparency and reproducibility of the evaluation, 2D images from the KITTI detection benchmark are used to train off-the-shelf camera-based detectors and evaluate the latter via a five-fold cross-validation. The provided 7481 non-sequential frames with ground-truth annotations are split in a 6:2:2 ratio and used for training, validation and testing respectively.

Table 4.1: Performance Evaluation of Deep Learning based 2D detectors

Model	Car	Pedestrian	Cyclist	Runtime
YOLOv3 (Redmon and Farhadi, 2018)	59.73%	38.51%	31.92%	21ms
Faster R-CNN (Ren et al., 2015)	81.57%	60.53%	57.83%	2635ms
EfficientDet (D1) (Tan et al., 2020)	75.83%	51.69%	46.92%	34ms

From the table, we can see that YOLOv3 shows the fastest processing speed among the three methods. However, this comes at the expense of object detection precision. In contrast, Faster R-CNN achieves the highest level of detection precision but requires longer single frame processing time. In contrast, the performance of EfficientDet represents a trade-off. This result led to EfficientDet being chosen as the off-the-shelf detector in the OTL implementation to better demonstrate online sample generation efficiency.

#### 4.4.2/ LEARN-BY-USE DETECTOR

The detector that needs to be learned online adopts a pipeline structure rather than an end-to-end one, since the former still has better interpretability than the latter. This feature is ideal in industrial applications for autonomous driving, where understanding the operation of the software is critical to the safety and reliability of the vehicle. Specifically, the implemented pipeline contains four modules:

- The **Ground Filtering** module first filters out points belonging to the ground in the point cloud generated by 3D lidar, making subsequent object segmentation more effective because the ground connecting them has been removed.
- The **Clustering** module performs segmentation on the filtered point cloud to generate clusters corresponding to different objects. The selection of the clustering

algorithm also considers the balance between segmentation accuracy and single frame processing time.

- The [Feature Extraction](#) module extracts feature vectors from clusters marked as learning samples.
- The [Online Classifier](#) module learns feature vectors and iterates itself over time.

The four modules in the pipeline work together to enable efficient and interpretable online model learning and road participant detection. Below we introduce their implementation in detail. For better explanation, let us first formalize the input point cloud as:

$$P = \{p_i \mid p_i = (x_i, y_i, z_i) \in \mathbb{R}^3, i = 1, \dots, I\} \quad (4.6)$$

#### 4.4.2.1/ GROUND FILTERING

Ground filtering is one of the basic steps in point cloud data processing in autonomous driving. Its two main purposes include reducing the computational burden (due to less points) of subsequent modules and improving the accuracy of point cloud segmentation. Commonly used ground filtering methods include threshold-based (Yan et al., 2017) and fitting-based (Zermas et al., 2017). The former is a simple method that removes points whose height (for example, along the z-axis) is lower than a preset threshold as the ground, which can be expressed as:

$$P^* = \{p_i \in P \mid z_i < threshold\} \quad (4.7)$$

This method is suitable for flat ground or horizontally installed lidar.

The calculation of the fitting method is relatively complex, but it is more robust to various terrains and sensor installation angles. Our system uses Ground Plane Fitting (GPF) proposed in (Zermas et al., 2017). Its working principle is as follows. The point cloud is first divided into segments evenly along the x-axis (vehicle driving direction). Then for each segment, a set of seed points with low height values are deterministically extracted to estimate the initial plane model of the segment's ground surface. This model is used to evaluate each point in the segment and generate the distance from that point to the orthographic projection on the candidate plane. Next, this distance is compared with a predefined threshold to determine whether the point belongs to the ground or not. Points belonging to the ground are used as seeds for a refined estimate of a new planar model, and the process is repeated a limited number of times. Finally, the entire ground is generated by concatenating the ground points extracted from each point cloud segment. The

entire process can be described formally as:

$$P^* = \{p_i \in P \mid \arg \min_{a,b,c} \sum_{i=1}^n |p_i - \hat{p}_i|\} \quad (4.8)$$

Figure 4.6 shows an example of using the GPF method to remove ground points. It can be seen that the proximal points can be effectively removed using this method, while the far points are still retained due to the smaller number of points that can be fit. Despite this, the GPF method is still considered a SOTA method at present.

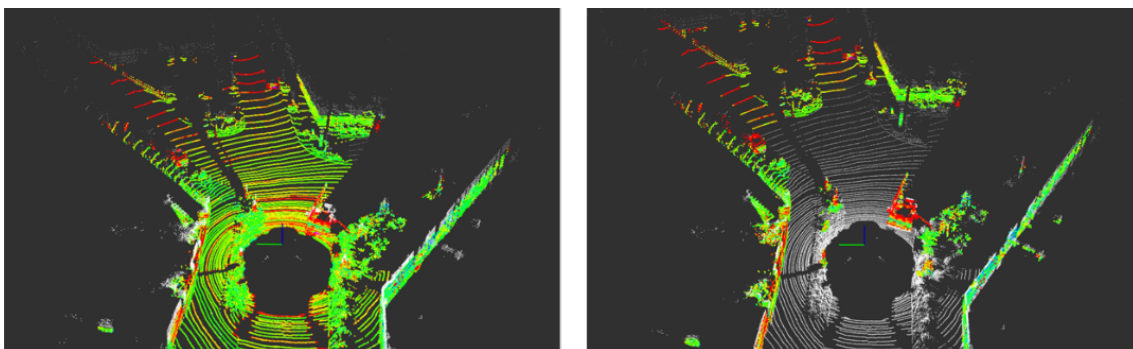


Figure 4.6: Schematic diagram of ground filtering. On the left is the original point cloud, and on the right is the result after using GPF. It can be seen that most of the points belonging to the ground are removed, and a small number of distant points representing slopes still exist.

#### 4.4.2.2/ CLUSTERING

The clustering method proposed by Autoware (Kato et al., 2018) is used to segment the non-ground point cloud obtained after ground filtering. Autoware is a ROS-based open source software stack for autonomous vehicles. It includes the enabling functions required for autonomous driving, from localization and object detection to route planning and control, and was created to enable as many individuals and organizations as possible to contribute to open innovations in autonomous driving technology. Consistent with Autoware's philosophy, and also to reflect the industrial friendliness of our implementation, the two key components in our system use modules provided by Autoware, one is clustering and the other is multi-target tracking. The former will be detailed in the immediately following paragraphs, while the latter will be introduced in Section 4.4.3.2.

To further reduce the computational burden, Autoware first projects the 3D point cloud onto a 2D plane:

$$P' = \lambda \cdot P^*, \lambda = (1, 1, 0) \quad (4.9)$$

However, it should be noted that this comes at the expense of clustering accuracy (Yang et al., 2022), since information in one dimension is masked after this step. Clustering is



therefore performed on the 2D plane, which is faster than in 3D space, and is defined as:

$$C_j \subset P', j = 1, \dots, J \quad (4.10)$$

where  $J$  is the total number of clusters. A condition to avoid overlapping clusters is that they should not contain the same points (Rusu, 2009), which is formalized as:

$$C_j \cap C_k = \emptyset, \text{ for } j \neq k, \text{ if } \min\|p_j - p_k\|_2 \geq d \quad (4.11)$$

where the sets of points  $p_j, p_k \in P'$  belong to the point clusters  $C_j$  and  $C_k$  respectively, and  $d$  is a Euclidean distance threshold. Finally, a volume filter is used to filter out those clusters that are too large or too small:

$$C^* = \{C_j \mid W_{min} \leq w_j \leq W_{max}, D_{min} \leq d_j \leq D_{max}, H_{min} \leq h_j \leq H_{max}\} \quad (4.12)$$

where  $w_j, d_j$  and  $h_j$  represent, respectively, the width, depth and height (in meters) of the volume containing  $C_j$ . Obviously, using a volume filter comes with a strong assumption: the size of the objects of interest is within the limits of the volumetric model. Unfortunately, this assumption doesn't always hold. However, with the current computing power of edge hardware, we must always make a compromise between accuracy and real-time performance. By filtering out a large number of background objects, the computational burden of subsequent modules can be alleviated. Furthermore, in all experiments reported in this dissertation, the thresholds for the volumetric model were set to:  $W = [0.1, 5.5]$ ,  $D = [0.1, 5.5]$  and  $H = [0.3, 5.5]$ . These are empirical values determined through multiple tests with the KITTI and Waymo datasets.

#### 4.4.2.3/ FEATURE EXTRACTION

Six features were extracted from the clusters, with a total of 61 dimensions, as summarized in Table 4.2, for subsequent classifier training. The set of feature values of each sample  $C_j$  forms a vector  $f_j = (f_1, \dots, f_6)$ . Features from  $f_1$  to  $f_4$  were introduced by (Navarro-Serment et al., 2009), while features  $f_5$  and  $f_6$  were proposed by (Kidono et al., 2011). The features are handcrafted, and their selection considers the trade-off between performance and dimensionality (Yan et al., 2020a). It should be noted that some other features are also proposed in (Navarro-Serment et al., 2009; Kidono et al., 2011; Yan et al., 2020a), and their performance is compared.

Table 4.2: 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

#### 4.4.2.4/ ONLINE CLASSIFIER

Recall that the proposed OTL is a learning method rather than a model. Therefore, in theory, any model can be used for OL, as long as it can meet the needs mentioned in Section 1.1. ORF is used because it is based on RF and therefore has the inherent advantages of fast training and support for multi-class classification. In addition, ORF, as an online extension of RF, proposes an online decision tree growing process, in which when to split a node depends on: 1) whether there are enough samples in the node for robust statistics, and 2) whether the split is good enough for classification. However, the original implementation of this method cannot be straightforwardly integrated into our OTL framework, and its binary tree-based data access performance still has room for improvement. We therefore conducted in-depth research on ORF and implemented corresponding improvements. More details and insights will be given in Chapter 5.

#### 4.4.3/ MULTI-TARGET TRACKER

In our general framework, the multi-target tracker is designed as a connector to connect different detectors, playing a key role in knowledge transfer and multi-source information fusion. The connection is loose, meaning that failure of one detector does not cause the failure of the entire system. In terms of implementation, our multi-target tracker contains two types of associations, one is spatial association involving low-level data fusion, and the other is temporal association involving high-level information fusion. It is worth pointing out that the fusion of separate low-level data or high-level information is common, while integrating the two into the same multi-target tracker in a spatio-temporal manner is not, the latter is thus novel. Below we describe the two association methods in detail.

#### 4.4.3.1/ SPATIAL ASSOCIATION

In the field of autonomous driving, calibration between various sensors and hard synchronization of the data they generate are two of the current de facto standard operations. Based on this fact, the 3D detection bounding box in the point cloud is first projected to the 2D image and then compared with the native 2D detection bounding box by measuring the Intersection over Union (IoU) to determine whether they are different detections of the same object. In all experiments reported in this dissertation, the thresholds for IoU were 0.7 for cars and 0.5 for pedestrians and cyclists, as per the settings of the KITTI benchmark suite. Once the IoU value exceeds the preset threshold, the two detections are associated. This step can be regarded as data fusion in different spaces at time  $t$ , which is summarized as Algorithm 1 for a more intuitive understanding.

---

#### Algorithm 1 Spatial association of detections for multi-target tracking

---

**Require:**  $O$ : 2D bounding boxes of the object detected in the image

$C^*$ : 3D bounding boxes of clusters in the point cloud

$\theta$ : IoU threshold

**Ensure:**  $\hat{C}^*$ : clusters associated with image-based detections

```

1: for each  $c \in C^*$  do
2:   Project  $c$  into the 2D image as  $c'$ 
3:   for each  $o \in O$  do
4:     if  $IoU(o, c') \geq \theta$  then
5:        $\theta \leftarrow IoU(o, c')$ 
6:        $c.associate(o)$ 
7:     end if
8:      $O.remove(o)$ 
9:   end for
10:   $\hat{C}^*.add(c)$ 
11: end for

```

---

#### 4.4.3.2/ TEMPORAL ASSOCIATION

On the other hand, detections in the same space at different times are associated. This step can be performed in a sequential or parallel manner with the aforementioned spatial association. In our implementation it is performed in a sequential manner following spatial association. In addition, it can be performed in any space such as 2D images or 3D point clouds, depending on which space the model is learned in. Our implementation uses Autoware's multi-target tracker, which tracks multiple objects over time based on Probabilistic Data Association (PDA), Unscented Kalman Filter (UKF), and Interactive Multiple Models (IMM).

The PDA algorithm updates the target state based on the weighted sum of the probabilities of each possible target state update. This algorithm is more robust to multi-target tracking under occlusion and clutter conditions than non-probabilistic methods such as the Nearest Neighbor (NN) method, but at the expense of consuming more computing resources. The UKF method is a combination of the Unscented Transform (UT) and the standard Kalman filter. The former enables the nonlinear system equations to be fitted to the latter under linear assumptions. This can effectively improve the accuracy and robustness of the tracking system.

The inclusion of IMM is one of the reasons we chose to integrate Autoware's tracker. It allows the construction of multiple motion models to perform combined estimation of the motion states of objects, including four steps of interaction, filtering, model probability update and combination of estimation. This design makes the tracker more robust to different categories of road participants. The state interaction among them reflects the essence of IMM. The process is describes as follows:

- **Interaction:** The input of the filter in model  $i$  is the combined estimation of each filter at the previous moment, and let the matching models at  $k - 1$  and  $k$  be  $j$  and  $i$ , respectively, then the equation of the combined probability is:

$$\mu_{k-1|k-1}^{ji} = \frac{1}{\hat{\mu}_{k|k-1}^i} \pi_{ji} \mu_{k-1}^j \quad (4.13)$$

where  $\mu_{k-1}^j$  is the probability of model  $j$  at  $k - 1$ ;  $\pi_{ji}$  denotes the probability that the object motion model transfer from model  $j$  to model  $i$ . The  $\hat{\mu}_{k|k-1}^i$  is defined as:

$$\hat{\mu}_{k|k-1}^i = \sum_{j=1}^n \pi_{ji} \mu_{k-1}^j \quad (4.14)$$

Then the re-initialized states and covariances were estimated as:

$$\tilde{x}_{k-1|k-1}^i = \sum_{j=1}^n \hat{x}_{k-1|k-1}^j \mu_{k-1|k-1}^{ji} \quad (4.15)$$

$$\begin{aligned} \tilde{P}_{k-1|k-1}^i &= \sum_{j=1}^n \mu_{k-1|k-1}^{ji} [\hat{P}_{k-1|k-1}^i + (\hat{x}_{k-1|k-1}^i - \\ &\tilde{x}_{k-1|k-1}^i) \cdot (\hat{x}_{k-1|k-1}^i - \tilde{x}_{k-1|k-1}^i)^T] \end{aligned} \quad (4.16)$$

- **Filtering:** The filtering using UKF filter includes prediction ( $P_{UKF}$ ) and update ( $U_{UKF}$ ), which are defined as:

$$[\tilde{x}_{k|k-1}^i, \tilde{P}_{k|k-1}^i] = P_{UKF}(\tilde{x}_{k-1|k-1}^i, \tilde{P}_{k-1|k-1}^i, f_{k-1}^i, Q_{k-1}^i) \quad (4.17)$$

$$[\hat{x}_{k|k-1}^i, \hat{P}_{k|k-1}^i] = U_{UKF}(\bar{x}_{k|k-1}^i, \bar{P}_{k|k-1}^i, z_k, h_k^i, R_k^i) \quad (4.18)$$

where  $f(\cdot)$  is the nonlinear state transfer function, and  $h(\cdot)$  is the nonlinear measurement function. Given the system process noise  $q_k \sim N(0, Q_{k-1})$  and the system measurement noise  $r_k \sim N(0, R_k)$ .

- **Model probability update:** Calculating the measurement likelihood function for each model:

$$\Lambda_k^i = \frac{\exp[-0.5(v_k^i)^T(S_k^i)(v_k^i)]}{\sqrt{|2\pi S_k^i|}} \quad (4.19)$$

where  $v_k^i$  and  $S_k^i$  are the measurement residuals and covariance of model  $i$ , respectively. The probability of model  $i$  at time  $k$  is defined as:

$$\mu_{k|k}^i = \frac{\mu_{k|k-1}^i \Lambda_k^i}{\sum_{j=1}^n \mu_{k|k-1}^j \Lambda_k^j} \quad (4.20)$$

- **Combination of estimation:** This step will calculate the combination of estimation and corresponding error covariance matrix at time  $k$ :

$$\hat{x}_{k|k} = \sum_{i=1}^n \mu_{k|k}^i \hat{x}_{k|k}^i \quad (4.21)$$

$$\hat{P}_{k|k} = \sum_{i=1}^n \mu_{k|k}^i [\hat{P}_{k|k}^i + (\hat{x}_{k|k}^i - \hat{x}_{k|k})(\hat{x}_{k|k}^i - \hat{x}_{k|k})^T] \quad (4.22)$$

#### 4.4.4/ SAMPLE ANNOTATOR

The sample annotator performs probabilistic fusion of samples associated with the multi-target tracker to determine in real time which class of objects the samples on each track belong to. Specifically, the Track Probability Yan et al. (2018) is applied, which is defined as follows. Let  $P(l_i|x_i, d_j)$  denote the probability that example  $x_i$  is an object with its category label  $l_i$  predicted by detector  $d_j \in D$  at the precise time  $t$ , then the track probability  $P(L_T|X_T, D)$  is computed by integrating the predictions of the different detectors according to the following formula:

$$P(L_T|X_T, D) = \frac{odds_{X_T}}{1 + odds_{X_T}} \quad (4.23)$$

where

$$odds_{X_T} = \prod_{i=1}^t \prod_{j=1}^K odds_{x_i}^j \quad (4.24)$$

and

$$odds_{x_i}^j = \frac{P(l_i|x_i, d_j)}{1 - P(l_i|x_i, d_j)} \quad (4.25)$$

By thresholding  $P(L_T|X_T, D)$ , high-confidence samples are finally labeled (i.e. car, cyclist, or pedestrian) and fed to the learn-by-use detector for learning. The threshold was set to 0.7 in our experiments.

## 4.5/ EXPERIMENTAL RESULTS

In this section, we evaluate the system implemented based on the general framework. In theory, evaluating the performance of sample generation should be to check how many learning samples generated online are consistent with the ground truth. However, since KITTI does not provide single-frame annotations of the raw data, we adopted an end-to-end system assessment strategy to conduct the evaluation from one side. The input end of the system is the raw sensor data, and the output end is the detection results of the road participants. All the experiments reported in this dissertation were performed with Ubuntu 18.04 LTS (64-bit) and ROS Melodic, with an Intel i9-9900K CPU and 64-GB RAM, and a NVIDIA GeForce RTX 2080 GPU with 8-GB RAM.

### 4.5.1/ DETECTION OF ROAD PARTICIPANTS

Object detection is the joint result of object localization and classification. In order to evaluate the performance of the learn-by-use detector, we use the test set (with ground truth) of KITTI and use the Average Precision (AP) (cf. Section 3.3.2) as the performance metric. The results are shown in the first row of Table 4.3. Furthermore, in order to verify the hypothesis that multi-modal detection may outperform single-modal detection, we evaluate the fusion performance of the point cloud-based learn-by-use detector and the image-based off-the-shelf detector, and the results are shown in the second row of Table 4.3. Specifically, they are the output of the “Sample Annotator” that has not undergone “Temporal Association”, since the test set consists of non-consecutive frames. It can be seen that with the participation of the image-based detector, the system’s 3D detection performance of various road participants has been significantly improved. This is due to the former’s more accurate classification based on color and texture information. In addition, this result also reflects the effectiveness of the spatial association algorithm we proposed (cf. Algorithm 1).

Table 4.3: Evaluation Results on The KITTI 3D Object Detection Validation Set

Method	Cars(%)			Pedestrians(%)		
	Easy	Moderate	Hard	Easy	Moderate	Hard
Learn-by-use detector	65.12	58.89	46.52	38.67	34.37	23.34
+ Off-the-shelf detector	<b>80.45</b>	<b>67.73</b>	<b>58.81</b>	<b>47.67</b>	<b>40.32</b>	<b>35.44</b>
Improvement	15.33	8.84	12.29	9	5.95	12.1

Method	Cyclists(%)		
	Easy	Moderate	Hard
Learn-by-use detector	50.73	34.14	26.67
+ Off-the-shelf detector	<b>75.61</b>	<b>56.55</b>	<b>47.61</b>
Improvement	24.88	22.41	20.94

Let's gain further insights through the performance improvement shown in the third row of Table 4.3. Among them, the detection of cyclists has the largest improvement (up to 24,88%), which can be explained by the fact that detecting this class in point clouds is more challenging than the other two, especially due to their irregular 3D geometric features and the intensity coming from various materials such as textiles, metals, rubber, etc. The smallest performance improvement occurs for pedestrians (up to 12,1%), due to their obvious 3D geometric features (i.e. mostly upright rectangular shapes) and the intensity mainly coming from textiles. It is also worth noting that, unlike the other two classes, the largest performance improvement in the pedestrian is for the "hard" difficulty. This can be explained by the fact that when pedestrians are occluded or truncated, the color and texture information are more helpful for detection compared to point clouds.

Table 4.4: Evaluation Results on The KITTI 2D Object Detection Validation Set

Method	Cars(%)			Pedestrians(%)		
	Easy	Moderate	Hard	Easy	Moderate	Hard
Off-the-shelf detector	82.16	78.18	67.09	60.25	49.97	44.84
+ Learn-by-use detector	<b>93.87</b>	<b>90.45</b>	<b>78.47</b>	<b>82.59</b>	<b>70.13</b>	<b>61.25</b>
Improvement	11.71	12.27	11.38	22.34	20.16	16.41

Method	Cyclists(%)		
	Easy	Moderate	Hard
Off-the-shelf detector	55.18	46.23	39.35
+ Learn-by-use detector	<b>80.25</b>	<b>75.31</b>	<b>62.87</b>
Improvement	25.07	29.08	23.52

On the other hand, we also obtained experimental evidence, as shown in Table 4.4, for the hypothesis that once the learn-by-use detector has competitive detection capabilities in

point clouds, it can even in turn help the off-the-shelf detector improve detection results on 2D images. Experimental results show that with the help of the point cloud-based learn-by-use detector, the overall performance of the system has been significantly improved. This is mainly due to lidar’s ability to detect distant objects, which is especially helpful for pedestrians cyclists and who are smaller than cars.

#### 4.5.2/ COMPARISON WITH OTHER METHODS

The test set (without ground truth) provided by KITTI for public ranking is used to compare our results with others in the community. Five representative methods are selected from KITTI’s ranking table for comparison. The principles of selection are: 1) methods using both image and point cloud data, 2) highly cited methods (MV3D (Chen et al., 2017), F-Pointnet (Qi et al., 2018), AVOD and AVOD-FRN (Ku et al., 2018)), and 3) recent methods (PFF3D (Wen and Jo, 2021)). The experimental results are shown in Table 4.5 and Table 4.6, respectively, where ours is the multimodal detector mentioned above.

Table 4.5: Evaluation Results Using Average Precision on The KITTI 3D Object Detection Test Set (for Ranking)

Method	Cars(%)			Pedestrians(%)		
	Easy	Moderate	Hard	Easy	Moderate	Hard
MV3D (Chen et al., 2017)	74.97	63.63	54.00	-	-	-
AVOD (Ku et al., 2018)	76.39	66.47	60.23	36.10	27.86	25.76
F-Pointnet (Qi et al., 2018)	<b>82.19</b>	69.79	60.59	<b>50.53</b>	42.15	38.08
AVOD-FRN (Ku et al., 2018)	82.11	71.70	67.08	50.46	<b>42.27</b>	<b>39.04</b>
PFF3D (Wen and Jo, 2021)	81.11	<b>72.93</b>	<b>67.24</b>	43.93	36.07	32.86
Ours	80.08	67.67	58.68	48.22	39.96	34.67
+ Size estimation	79.97	69.13	58.57	48.65	40.11	35.99

Method	Cyclists(%)		
	Easy	Moderate	Hard
MV3D (Chen et al., 2017)	-	-	-
AVOD (Ku et al., 2018)	57.19	42.08	38.29
F-Pointnet (Qi et al., 2018)	72.27	56.12	49.01
AVOD-FRN (Ku et al., 2018)	63.76	50.55	44.93
PFF3D (Wen and Jo, 2021)	63.27	46.78	41.37
Ours	<b>73.72</b>	<b>56.64</b>	47.94
+ Size estimation	<b>75.20</b>	<b>58.96</b>	<b>50.41</b>



Table 4.6: Evaluation Results Using Average Precision on The KITTI 2D Object Detection Test Set (for Ranking)

Method	Cars(%)			Pedestrians(%)		
	Easy	Moderate	Hard	Easy	Moderate	Hard
MV3D (Chen et al., 2017)	93.08	84.39	79.27	-	-	-
AVOD (Ku et al., 2018)	95.17	89.88	82.83	50.90	39.43	35.75
F-Pointnet (Qi et al., 2018)	95.85	<b>95.17</b>	85.43	<b>89.83</b>	<b>80.13</b>	<b>75.05</b>
AVOD-FRN (Ku et al., 2018)	94.70	88.92	84.13	67.95	57.87	55.23
PFF3D (Wen and Jo, 2021)	95.37	92.15	<b>87.54</b>	62.12	52.53	50.27
Ours	93.11	91.20	78.22	83.08	69.97	60.98
+ Size estimation	<b>96.31</b>	91.17	81.20	84.74	71.45	64.58

Method	Cyclists(%)		
	Easy	Moderate	Hard
MV3D (Chen et al., 2017)	-	-	-
AVOD (Ku et al., 2018)	66.45	52.60	46.39
F-Pointnet (Qi et al., 2018)	<b>86.86</b>	73.16	65.21
AVOD-FRN (Ku et al., 2018)	70.38	60.79	55.37
PFF3D (Wen and Jo, 2021)	79.44	66.25	60.11
Ours	81.99	<b>74.90</b>	63.06
+ Size estimation	85.62	<b>76.88</b>	<b>66.04</b>

As can be seen from Table 4.5, our online learning framework exhibits competitive performance compared with other offline methods, especially in the class “cyclists” we achieved the best results in easy and moderate difficulty. On the other hand, although our method fails to achieve the best results on the two classes of “car” and “pedestrian”, the gap is not significant. Taking moderate difficulty as an example, ours is 5.26% and 2.31% away from the best performers, respectively. The reason, on the one hand, is that the clustering method we use suffers from the accurate segmentation of road participants, which leads to some missed detections and false alarms. On the other hand, if some clusters are too small in the 3D point cloud, they will be ignored due to their low IoU value after being projected into the 2D image.

Correspondingly, Table 4.6 shows the object detection performance of our multimodal detector on 2D images in comparison with other methods. It can be seen that, for moderate difficulty, ours still wins on the cyclist detection, and is only 3.97% away from the best performance on car detection. As for pedestrian, although we are 10.16% behind the first place, we are still in the top 7% of the entire KITTI ranking.

### 4.5.3/ SIZE ESTIMATION

KITTI’s object annotations in point clouds are based on projections from those in 2D images, thus including an estimate of the size of the object in its 3D bounding box. In contrast, point cloud clustering methods commonly used in autonomous driving, such as the Autoware method we use, output the true size of the clusters. Based on this fact and in order to present a fairer comparison with other methods, we additionally provide the results of our multimodal detector after adding size estimation in Table 4.5 and Table 4.6. Specifically, we randomly sample the training set to estimate volumetric models for each object class, and if a cluster is classified, then the convex box representing its size is inflated to the bounding box defined by its volume model. An intuitive result is shown in Figure 4.7.

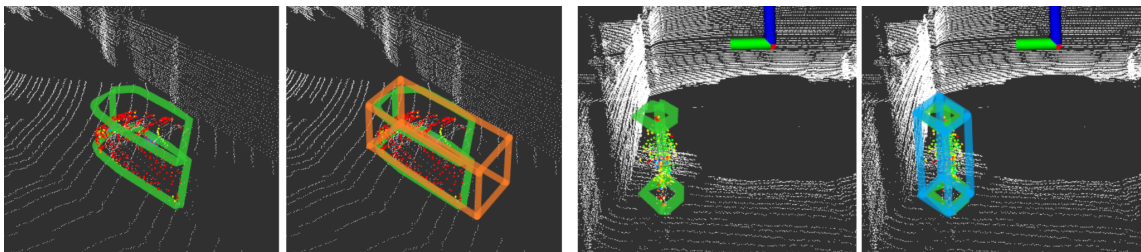


Figure 4.7: Schematic illustration of 3D object size estimation in point clouds. Through size estimation, the original convex box is inflated to the bounding box. The left two images show a car, while the right two show a pedestrian.

As can be seen from Table 4.5 and Table 4.6, after adding size estimation, the detection performance of each class of objects shows an overall improvement, which is especially obvious in the detection of cyclists. This is mainly due to the special structure of cyclists, they usually contain a vertical person and a horizontal bicycle, so their bounding boxes defined in the point cloud by polygons are usually smaller than the manual annotations provided by KITTI. Furthermore, it is worth pointing out that if the test set consists of consecutive frames, then the “Temporal Association” module in our multi-target tracker can be used to estimate the true size of objects, thus sidestepping the need for empirical volumetric models.

### 4.5.4/ ABLATION STUDY

The ablation experiment aims to study the contribution of each module in a system to the overall performance of the system. In our case, the classification performance of detectors based on different system architectures is evaluated after each iteration in the online learning process using the Macro Average (MaA) metric. The experimental results are shown in Figure 4.8. It can be seen that simply introducing 2D image detection to match

point cloud clusters in 3D space can make the online learning performance present an overall upward trend (shown by the green line), but the improvement is not obvious after the environment becomes complex (starting from about 1000 frames, more pedestrians and cyclist appear). This situation is improved by introducing temporal association to the samples (shown by the blue line).

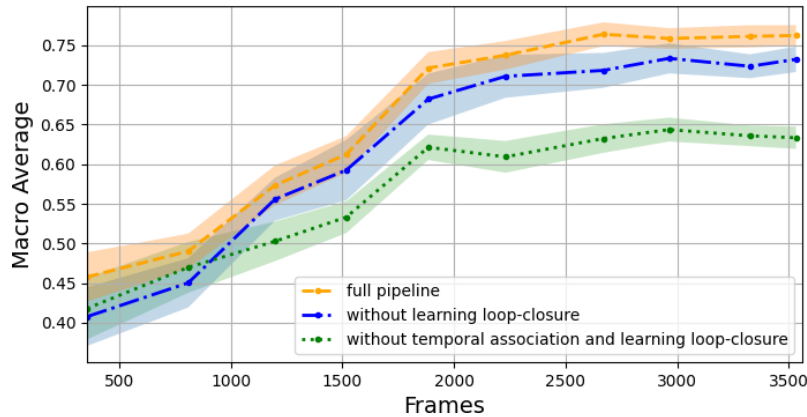


Figure 4.8: Ablation experiment results.

In addition, the classification performance of the detector is also improved overall. This is mainly because the temporal association makes those clusters that are not matched by the spatial association likely to become learning samples later, thus improving the learning efficiency. In addition, by introducing the learning loop-closure, i.e., the probability estimation of the learn-by-use detector to the class of the point cloud cluster is also input into the multi-target tracker and then becomes a part of the learning, so that the online learning performance is improved again, as shown in orange line.

## 4.6/ QUALITATIVE ANALYSIS

It can be seen from Fig. 4.9 that almost all road participants are correctly detected in the matched image and point cloud and tracked in the latter. The section highlighted by the blue box in the point cloud demonstrates that even though the vehicle is no longer within the camera's field of view, it is still a learn-able sample being tracked. And the range box area indicates that despite being detected as a cyclist in the image, the object is correctly learned as a pedestrian by the classifier through spatio-temporal association. While one exception is the part annotated by the red box, where the cyclist and the utility pole are too close together, leading to their aggregation as a single entity (i.e., under-segmented), resulting in the failure of correct detection. This is a challenge both visually and in 3D lidar. One possible solution to improve this issue is to incorporate color and texture information into the point cloud using pixel-to-point matching for further segmentation.

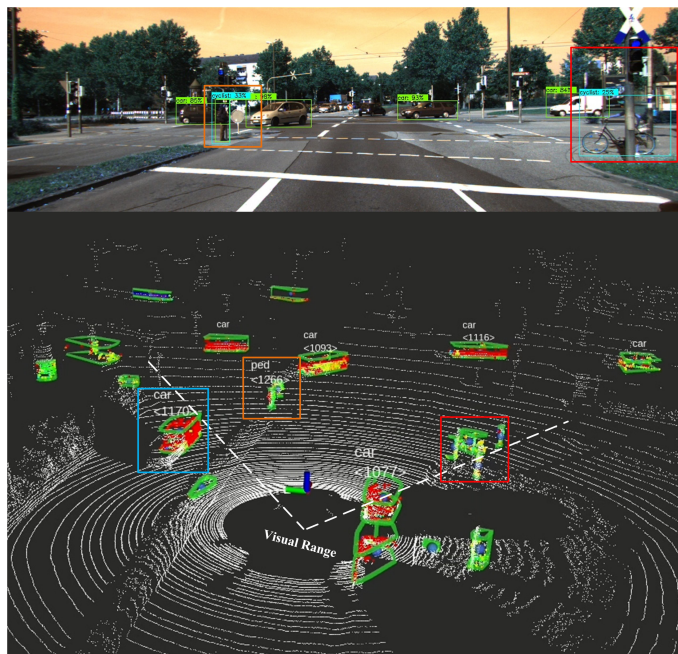


Figure 4.9: A synchronized pair of images and point clouds from the KITTI dataset. The labels in the images contain the class information, while the labels in the point clouds contain both the class and Track-ID. The colored points enclosed by labels and bounding boxes represent the samples used for online learning.

For the segmentation of the point cloud data, in general, the segmentor shows good performance in both simple (upper part of Fig.4.10) and complex (lower part of Fig.4.10) 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 part of Fig.4.10, people and benches are clustered as a whole. Followed with the fusion detection part, the first is the performance of the 2D detector that plays the role of the trainer. As can be seen from lower part of Fig.4.10, the image detection still shows acceptable performance, and the matching with point cloud cluster is also robust. Most of the objects detected in the image can be fully fitted. The second is the role of the multi-target tracker. It can be seen from the upper part of Fig.4.10 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 part of Fig.4.10. This undoubtedly greatly improves the learning efficiency of the entire system.

Fig.4.11 displays an detection result example of our system, demonstrating a relatively complex scene with several occluders and a large number of background objects. Objects visible in the visual range are given by 2D bounding boxes on the image, and objects in the point cloud are indicated by 3D bounding boxes. A similar situation occurs when pedestrians are gathered, for example, in the near and far crowds shown in the figure.

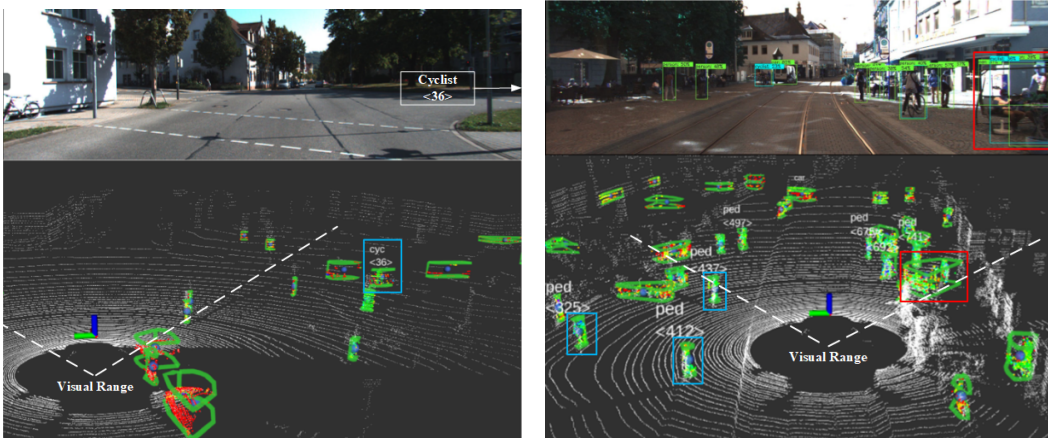


Figure 4.10: Two pairs of synchronized frames include the color image and its corresponding 3D lidar scan on KITTI. The coordinate axis in the point cloud represents the position of the 3D LIDAR sensor. The colored point cloud enclosed by the bounding box represents the samples input to the classifier. Note that only those labelled samples will be learned by the classifier, such as the cyclist in the upper image. The upper and lower parts respectively show the performance of our system in simple and complex scenarios.

In the point cloud that on the right side of the illustration, we can see the presence of a large number of columnar objects in the scene, which are usually difficult to distinguish them from pedestrians and can be effectively solved by 2d detection. In the image we can see that the pedestrians in the middle of the slightly distant crowd actually obscured by the lamp post, while the vehicles in their vicinity are also too far away to be clearly identified, making them visually difficult to be detected, but they can be accurately segmented and detected in the point cloud, which can help improve the performance of 2d detection on the other hand. Meanwhile, it is noticed that other objects outside the visual range can also be detected accurately by our fusion detection.

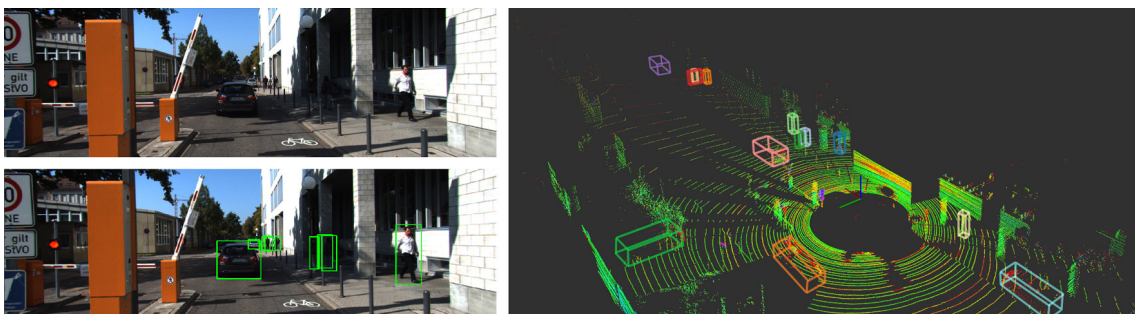


Figure 4.11: A example of the detection results on the KITTI data. The left side represents the final result of 2D detection, and the right side represents the final result of 3D detection.

## 4.7/ CONCLUSION

In this chapter, the question of how to efficiently generate samples for online learning in point clouds is answered. We first formulated the problem as an Online Transfer Learning problem associated with an Information Fusion problem, based on the fact that 1) cameras are already one of the standard sensors in autonomous vehicles, and that 2) they can provide reliable object detection in clear weather conditions, thanks to their ability to capture color and texture information from the environment. We then introduce a general solution framework for the posed problems. The framework was originally developed in the field of mobile robotics and then perfectly transplanted to the field of autonomous driving in the context of this dissertation. It contains four main modules, namely Off-the-shelf Detector ( $D_{off}$ ), Learn-by-use Detector ( $D_{learn}$ ), Multi-target Tracker ( $T_{multi}$ ), and Sample Annotator ( $A_{sample}$ ). Among them,  $D_{off}$  mainly plays the role of a trainer to guide the learning of  $D_{learn}$  and help it grow.  $T_{multi}$  is the communication channel between the trainer and the learner, realizing the transfer of knowledge from  $D_{off}$  to  $D_{learn}$  over time. Finally,  $A_{sample}$ , who plays the role of discriminator, supervises the interaction between  $D_{off}$  and  $D_{learn}$  and resolves conflicts.

We then proposed a system-level implementation of the introduced framework that is tailored for autonomous driving and includes a novel spatio-temporal information fusion mechanism for  $T_{multi}$ . Specifically  $D_{off}$  is implemented as an image detector based on deep learning, and  $D_{learn}$  is implemented as a point cloud detector based on Online Random Forest.  $T_{multi}$  first finds different detections of the same object at the same time in two different spaces, the image and the point cloud, and then further correlates the detections of the same object at different times in the point cloud space.  $A_{sample}$  finally uses odds to perform probability fusion on the detection information associated with each object to provide the learning samples required by  $D_{learn}$ .

To illustrate the effectiveness of our proposed method, extensive experiments are conducted on the KITTI dataset. Through the performance evaluation of the online learning model for the detection of road participants including cars, pedestrians and cyclists in the urban environment, on the one hand, the quality of the samples generated online is evaluated from the side, and on the other hand, it is shown that our method enables 3D lidar to quickly learn the detection ability of road participants without ground truth and human intervention. What is even more interesting is that the experimental results show that both  $D_{off}$  and  $D_{learn}$  improve the (multi-modal) detection performance of the overall system in their respective spaces by helping each other. Finally, the ranking achieved by submitting the experimental results to the KITTI benchmark shows that our method is competitive in both 2D and 3D object detection performance, especially in the categories of pedestrians and cyclists, surpassing most existing methods.



# 5

## EFFICIENT ONLINE MODEL TRAINING AND ACCESS



## 5.1/ INTRODUCTION

According to the definition given in Chapter 1, Online Learning (OL) is a model training method rather than a model. Any model can be integrated into the OL method, as long as it meets the following requirements: 1) it can be quickly trained, 2) it can be deployed immediately, and 3) it can be updated at any time. For autonomous driving, an additional requirement (denoted as “Requirement 4”) is that the model should be explainable, for example through mathematical modeling. This is particularly important for the deployment of models in industrial products because it involves system certification issues. Faced with the above four requirements, the community currently lacks convincing solutions.

Commonly used machine learning models can be roughly divided into two categories. One is traditional methods such as Support Vector Machine (SVM), Adaptive Boosting (AdaBoost), Random Forest (RF), etc., while the other is deep learning methods. Previous work (Broughton et al., 2020) showed that the real-time performance achieved by integrating deep learning methods comes at the expense of model performance. Furthermore, deep learning methods currently do not meet Requirement 4. Therefore, traditional methods are the main focus of this dissertation. Previous work integrated SVM into the OL method (Yan et al., 2017), but it solved a binary classification problem, which is difficult to meet the detection needs of multi-class road participants in autonomous driving. Besides, existing literature does not show the integrability of AdaBoost due to its inability to meet Requirement 1.

As mentioned in Chapter 4, the Online Random Forest (ORF) is integrated into our OL framework as a training model. This method is essentially a RF method, therefore meets Requirements 1 and 4. On the other hand, as it can update the model with incoming data (that is, online, which is different from batch training methods), it partially meets Requirement 3. Note that the “partial satisfaction” mentioned here is because the ORF has an important assumption that the amount of data and their distribution are known before model training, so that the ORF can calculate the optimal training parameters. This obviously does not meet the needs of autonomous driving. Besides that, the ORF does not satisfy Requirement 2. To solve these problems, in our research, the ORF is first ROSified, and then three features are added including support for streaming data, few-shot training without knowing the total amount and the distribution of data, and real-time model access, making it to be smoothly integrated into our OL framework.

Based on the integration of the ORF, we are also interested in how to push the access performance of the model to a new boundary, including model size, access time, and accuracy of knowledge preservation. Specifically, inspired by OctoMap (Hornung et al., 2013), an efficient point cloud data representation widely used in the robotics community, we replace the binary trees used in the ORF with octrees, and conduct a comparative

analysis of the access performance of the models before and after the replacement. It should be pointed out that our research is not intended to substitute the widely used deep learning-based model in the current industrial context of autonomous driving, but to run in parallel with it in a dual-model manner. Scientifically, we are full of expectations for the future integration of deep learning methods that meet Requirements 1-4 in OL methods.

## 5.2/ PROBLEM FORMULATION

### 5.2.1/ EFFICIENT MODEL TRAINING

In the context of OL, efficient model training can be formalized as a local optimization problem. Given one or several learning data, the goal is to find the optimal parameters for model training:

$$\arg \min_{\theta} \text{Loss}(\theta) = \left\{ \sum L(y_i, f_{\theta}(x_i)) \right\} \quad (5.1)$$

where  $\theta$  represents the parameter to be optimized,  $x_i$  represents the features of the  $i$ -th sample,  $y_i$  represents the true labels of the  $i$ -th sample, and  $f_{\theta}(x_i)$  denotes the prediction of the online model. Specifically, in ORF,  $\theta$  contains: 1) the minimum number of samples a node has to see before splitting  $\alpha$ , 2) the minimum gain a split has to achieve  $\beta$ .

### 5.2.2/ EFFICIENT MODEL ACCESS

Efficient model access can be formalized as a global optimization problem, that is, given a certain amount of data, how to minimize the space for storing them:

$$\arg \min_{\phi, \lambda} \text{Space}(\phi, \lambda) = \left\{ \frac{1}{\phi} \sum_{i=1}^{\phi} \text{Data} + \lambda \cdot \text{Regularization} \right\} \quad (5.2)$$

where  $\phi$  represents the parameters used to optimize data storage (if  $\phi$  is 1, the compression is lossless),  $\lambda$  is the regularization parameter, Data represents the storage space of the data, and Regularization is the regularization optimization terms, which can be defined according to specific problems and used to balance the trade-off between storage space and other optimization goals (e.g. time and accuracy).

## 5.3/ METHODOLOGY

Methodologically, we are based on the ORF, mainly use a few-shot training strategy to respond to the problem defined in Section 5.2.1, and use octrees to respond to that defined in Section 5.2.2. Three subsections are used to introduce them in detail below.

Since the other two improvements mentioned in the Section 5.1, i.e. support for data streaming and real-time model access, are mainly engineering, please refer to the code we released for details.

### 5.3.1/ ONLINE RANDOM FOREST

It is essential to recall the concept of the ORF first. The classic RF algorithm uses batch training to generate decision trees based on global data, which does not meet the needs of OL. In contrast, the ORF proposes an online decision tree growing procedure where when to split a node depends on: 1) whether there are enough samples in the node to have robust statistics, and 2) whether the split is good enough for classification purposes. This idea is further concreted as:

$$|R_j| > \alpha \quad \wedge \quad \exists s \in S : \Delta L(R_j, s) > \beta \quad (5.3)$$

where  $\alpha$  is the minimum number of samples a node needs observe before splitting,  $\beta$  is the minimum gain that needs to be achieved for node splitting,  $R_j$  is a decision node, and  $\Delta L(R_j, s)$  is the gain of node  $j$  with respect to the test  $s$ , which is measured as:

$$\Delta L(R_j, s) = L(R_j) - \frac{|R_{jls}|}{|R_j|} L(R_{jls}) - \frac{|R_{jrs}|}{|R_j|} L(R_{jrs}) \quad (5.4)$$

where  $R_{jls}$  and  $R_{jrs}$  are respectively the left and right partitions based on  $s$ . The idea is to find the test with the highest gain as a node splitting decision:

$$s_j = \operatorname{argmax}_{s \in S} \Delta L(R_j, s) \quad (5.5)$$

### 5.3.2/ FEW-SHOT TRAINING

As mentioned before, the original ORF needs to know the total number of data and their distribution to perform global feature selection, and then updates the model based on single sample. But if we invalidate the assumption that the ORF has a prior on the data, then the performance of model updates based on such global information is not guaranteed for data with unknown distributions. Moreover, by leverage the power of the spatio-temporal association method presented in Section 4.4.3, that is capable of correlating different detections belonging to the same object, to empower the ORF to calculate the optimal training parameters via statistical analysis by only considering learning samples generated in an observation time window of a sensor (such as the 3D LiDAR). Therefore, we propose to use a few-shot training strategy. More specifically, the ORF still maintains model updating based on single samples, but based on a local optimal strategy.

Exponential Moving Average (EMA) is employed to dynamically update the average value of each feature in the learning sample, enabling localized optimization in the case of continuous data input into the classifier, while old data is not retained. It assigns a higher weight to new data while gradually decreasing the weight of old data, thus allowing it to adapt to changes in data distribution. The update process of EMA is as follows: For each new data point, the calculation of EMA for each feature is as follows:

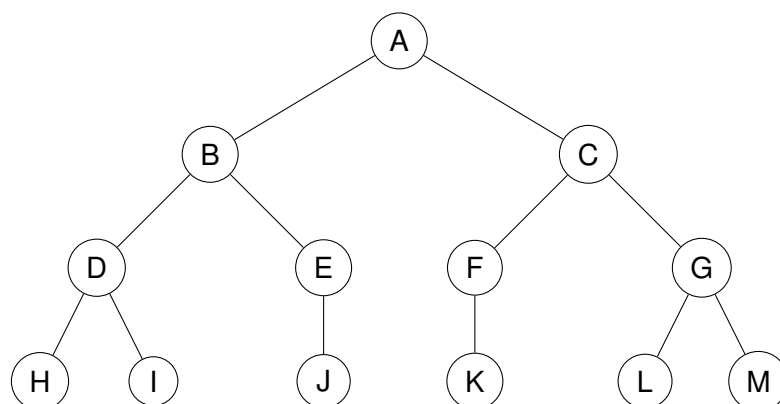
$$E_t = (1 - \alpha) * E_{t-1} + \alpha * NewData_t \quad (5.6)$$

Where  $E$  is the current value of the feature,  $\alpha$  is the smoothing factor, typically taking a value between 0 and 1, controlling the weight of the new data point. A higher  $\alpha$  value gives more weight to the new data, making  $E$  more responsive to changes but less stable.

EMA gradually adjusts the value of the corresponding feature according to the new sample, giving a higher weight to new data, but also retains a certain weight of past data, so it can adapt to the changes.

### 5.3.3/ OCTREE

With Continual Learning (CL) as the background, driven by reducing the size of a model as well as the time-consuming of reading the model, we propose to use the octree to replace the binary tree originally used in the ORF, as the former can provide a more effective representation to complex data than the latter, although this representation for knowledge preservation may be performed in a lossy way (insights will be given in Section 5.4.3). Specifically, as shown in the upper part of Figure 5.1, a node of a binary tree usually has only two child nodes including a left and a right child nodes. This tree structure is simple, but has higher access cost than other tree structures. In contrast, the octree, as shown in the middle of Figure 5.1, contains eight child nodes, which can effectively reduce the storage of redundant information (intermediate nodes), but at the expense of the possibility of losing useful information.



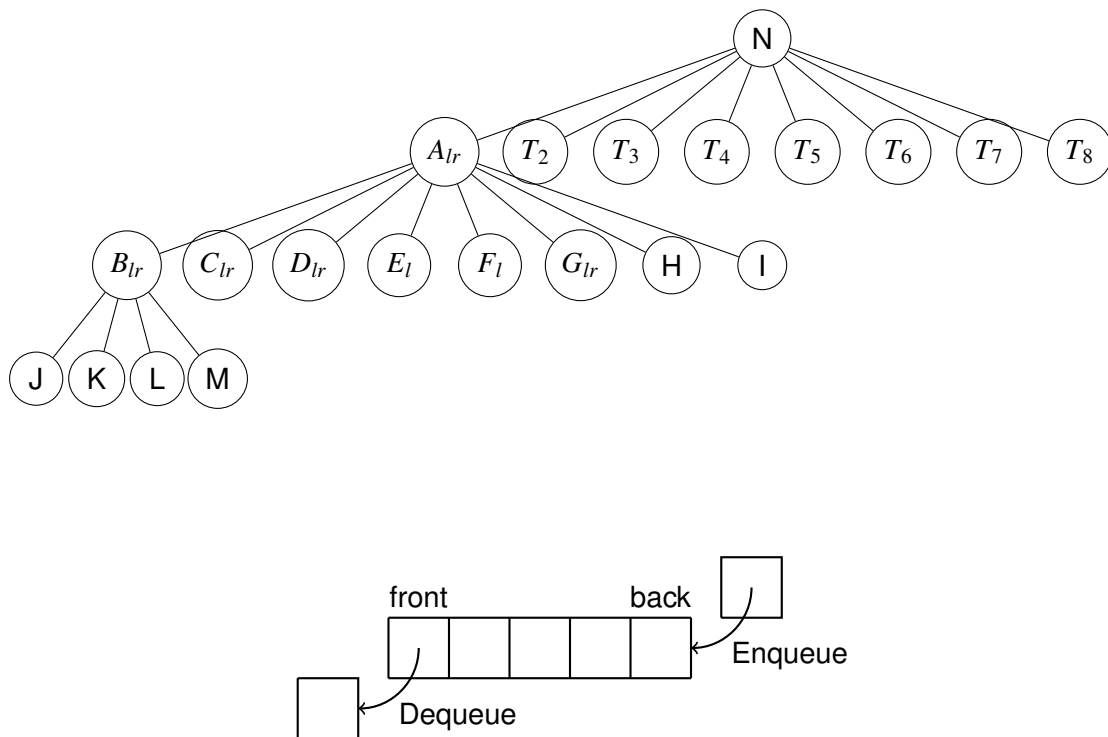


Figure 5.1: Binary tree (top), octree (middle), and FIFO query mechanism (bottom)

Therefore, a level-order traversal based on First In First Out (FIFO) query (as shown in the lower middle part of Figure 5.1) for each decision tree is performed, then the node information of the binary tree is transferred to the octree, and finally the latter is used as the model access structure of the ORF. The details about the transfer of node information are summarized in Algorithm 2.

## 5.4/ EXPERIMENTAL RESULTS

The experiments mainly include three aspects: 1) To confirm that the ORF is suitable for OL of road participant detection in point clouds, the original ORF is compared with the classic RF; 2) To evaluate our improvements, the improved ORF is compared with the original one; 3) To evaluate the octree, the performance of model access using octrees is compared to that using binary trees. In addition, corresponding to the main research motivation of this chapter, i.e. Online Continual Learning (OCL), experiments across datasets are conducted, namely from KITTI to Waymo. Details are given in the remainder of this section.

---

**Algorithm 2** Information transfer from binary tree to octree based on level-order traversal
 

---

**Require:** BINARYTREE  $*root$ , OCTREE  $*node$ **Ensure:** OCTREE

```

1: for each children  $i \in I$  do
2:   if  $node.Children[i]$  is empty then
3:      $node_{empty} \leftarrow 1$ 
4:      $node \leftarrow node.Children[i]$ 
5:     break
6:   end if
7: end for
8: if  $node_{empty}$  is not 1 then
9:   CreatOctree(OCTREE  $*newnode$ )
10:   $node \leftarrow newnode.Children[i]$ 
11: end if
12:  $queue \leftarrow$  empty queue,  $state \leftarrow 0$ 
13: enqueue( $queue$ ,  $root$ )
14: while  $queue$  is not empty do
15:   $root \leftarrow$  dequeue( $queue$ )
16:  if  $root.left$  is not null then
17:    enqueue( $queue$ ,  $root.left$ ),  $state \leftarrow 1$ 
18:  end if
19:  if  $root.right$  is not null then
20:    enqueue( $queue$ ,  $root.right$ )
21:    if  $state$  is 0 then
22:       $state \leftarrow 2$ 
23:    else
24:       $state \leftarrow 3$ 
25:    end if
26:  end if
27:  InsertToOctree( $root$ ,  $state$ ),  $state \leftarrow 0$ 
28: end while

```

---

#### 5.4.1/ ONLINE RANDOM FOREST VS. RANDOM FOREST

To assess the performance of the native ORF, a subset of 1500 samples (i.e. 60% of the 7481 point cloud frames) was randomly selected from the training set of KITTI to form a data stream as the input of the ORF. Whenever the ORF learns 100 samples, we save a model and evaluate it on the test set with ground truth (i.e. 20% of the 7481 point cloud frames). We report results for the first 15 iterations. In parallel, a conventional RF,

trained offline with Scikit-Learn (Pedregosa et al., 2011) using the same batch of 1500 samples, was employed as a baseline for comparative analysis. Both the online and offline approaches adhered to a uniform parameter configuration for the forest structure, specifically,  $trees = 100$ ,  $depth = 50$ ,  $epochs = 20$ ,  $split\_threshold = 50$ , which ensured a consistent basis for evaluating the performance of the two methods. For the settings of other parameters, please refer to the code we released. The outcomes of the experiments are depicted in Figure 5.2.

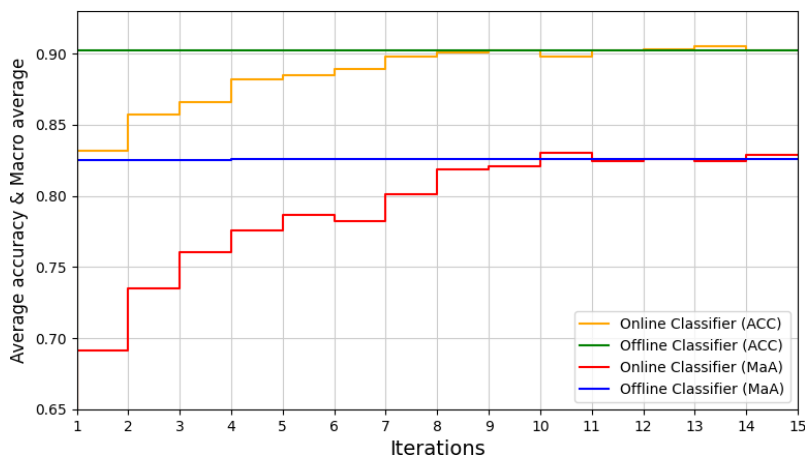


Figure 5.2: Performance comparison of the ORF-based (online-trained) and the RF-based (offline-trained) 3D LiDAR-based object classifiers.

It can be seen that as the quantity of training samples increases, the performance of the ORF swiftly converges with that of the offline-trained counterpart. This convergence is particularly notable after the 8 – 9<sup>th</sup> iteration and tends to be stable, as indicated by both Average Accuracy (ACC) and Macro Average (MaA) metrics.

#### 5.4.2/ INTEGRATED-ORF VS. STANDALONE-ORF

Since our improvements to the ORF are mainly to make it comply with our definition of OL requirements in the autonomous driving domain, it is a matter of course that the experimental evaluation uses the entire OL framework introduced in Chapter 4 (cf. Figure 4.5). Specifically, the original ORF is named “Standalone-ORF” in the experiment because it was independently trained with ground-truth annotations. In contrast, the improved one is called “Integrated-ORF” as it was learned autonomously without ground-truth annotations (i.e. with KITTI’s raw data) within the OL framework. For Standalone-ORF, the same 1500 samples as introduced in Section 5.4.1 was used. For both, the training is iteratively performed every 100 samples obtained, and their performance is evaluated after 15 iterations with the same test set as per Section 5.4.1. Both ORFs are consistent in the parameter setting of the forest structure also as per Section 5.4.1. The evaluation

compares their classification performance on cars, pedestrians and cyclists. The results are represented using confusion matrices, as shown in Figure 5.3.

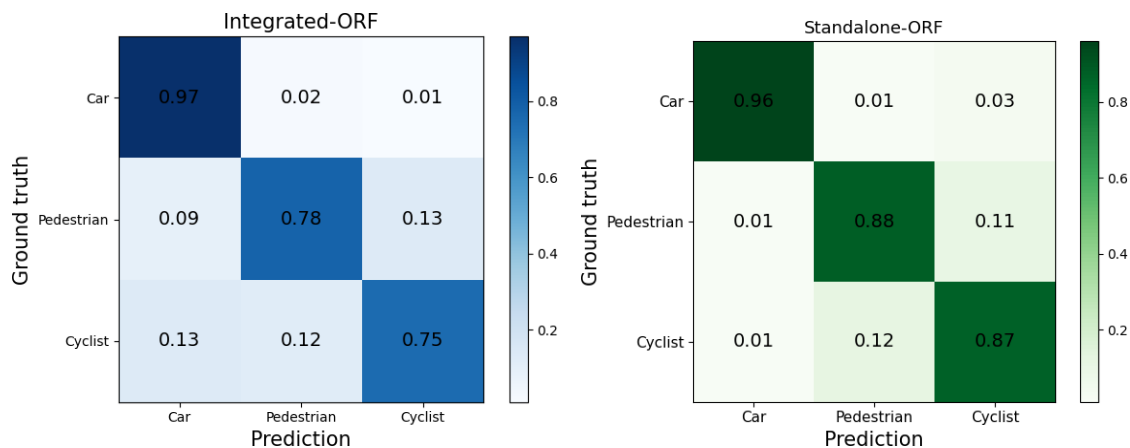


Figure 5.3: Confusion matrix for performance comparison of Integrated-ORF and Standalone-ORF. The ordinate represents the true label, and the abscissa represents the predicted result. The darker the color, the higher the proportion of correct classifications.

It can be seen that the performance of Integrated-ORF does not catch up with Standalone-ORF in an all-round way, and the gap is concentrated in the classification of cyclists and pedestrians, which are the challenges in the perception of autonomous driving. In contrast, Integrated-ORF outperformed Standalone-ORF even slightly on car prediction. There are two factors for this result. One is that the camera-based detector that plays the role of “trainer” has different detection capabilities for different object categories (cf. Table 4.1) Note that we use a multi-class detector instead of some of the top-ranked single-class detectors on KITTI. The second is that the point cloud clustering method based on unsupervised learning is currently not up to the accuracy of manual segmentation, especially when it also includes projecting the point cloud in 3D space to a 2D plane (i.e. trading precision for speed), making it difficult for relatively small size objects to be accurately segmented (Yan et al., 2020a). These two reasons cause Integrated-ORF to learn some false positive samples. However, it is still worth pointing out that Integrated-ORF is learned without ground truth and human intervention.

### 5.4.3/ OCTREE VS. BINARY TREE

In order to evaluate the performance of the octree, the original binary tree structure access model in the Integrated-ORF was replaced with the octree structure, and the experiment was run again under the exact same experimental settings. The difference is that only the results of the first 10 iterations are reported (due to the results are close to convergence after 8 iterations). The experimental results are shown in Figure 5.4.



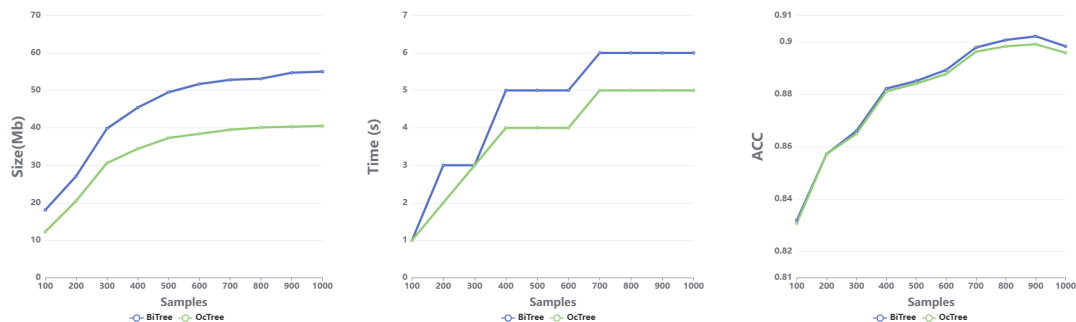


Figure 5.4: Comparison of model access performance based on binary trees and octrees.

It can be seen that octree-based model access has obvious advantages over binary trees in terms of model size (c.f. the left in Figure 5.4) and model loading time (c.f. the middle in Figure 5.4). However, as shown on the right in Figure 5.4, this comes at the expense of model accuracy. In other words, the current method we propose cannot guarantee that the octree will surpass the performance of the binary tree in terms of knowledge preservation with the iteration of learning. This is also the reason why we did not continue to use octrees in subsequent research (i.e. Chapter 6). A potential solution could be to develop node splitting and tree update strategies tailored to ORF based on the structural characteristics of the octree.

#### 5.4.4/ LEARN ON KITTİ TEST ON WAYMO

In fact, the study of model training and access is to establish the basis for extending OL to Online Continual Learning (OCL), which experimentally corresponds to our idea of cross-dataset evaluation. To this end, the Integrated-ORF learned in KITTİ is deployed to the Waymo dataset and is online updated. The selection of learning data is as described in Section 3. The experimental results evaluated using the MaA are shown in Figure 5.5.

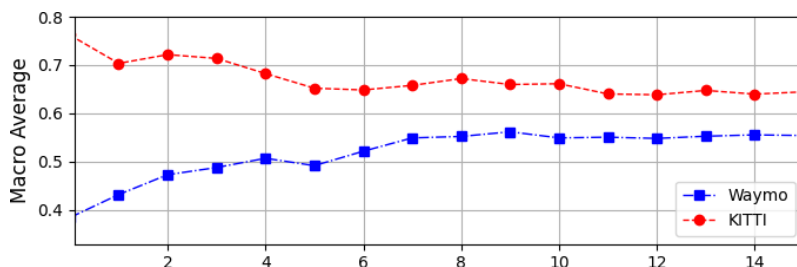


Figure 5.5: Changes in classification performance after deploying the Integrated-ORF learned on the KITTİ dataset to the Waymo dataset and continuing to learn. The blue line represents the results evaluated on the Waymo test set. The red line indicates the results evaluated on the KITTİ test set.

It can be seen that due to the significantly different driving scenarios exhibited by the

KITTI and Waymo datasets, the classification performance of the Integrated-ORF shows a cliff-like decline at the beginning. However, thanks to the OL framework, with the input of Waymo data, the classification performance of the Integrated-ORF improves steadily with the learning iterations and finally stabilizes after the 8 – 9<sup>th</sup> iteration. On the other hand, evaluating the updated model with the Waymo dataset back to the test set of KITTI found that its performance overall decreased with learning iterations. This is a manifestation of catastrophic forgetting, which will be the focus of the next chapter.

The final classification performance of the detector after 15<sup>th</sup> iterations is shown in Figure 5.6. It can be seen that the overall performance has declined compared to the one reported in Section 5.4.2, where the drops are 0.05, 0.09, and 0.15 for cars, cyclists, and pedestrians, respectively. This is primarily due to the fact that if the scenes selected from Waymo are simple, the Integrated-ORF learned on the KITTI dataset performs admirably, but it fails to effectively demonstrate the strengths of the OL framework. Therefore, the selected clips containing complex scenes pose a huge challenge to the segmentation of the point cloud and the object tracking algorithm in it, which directly affects the quality of the learning samples. However, it is gratifying that the experimental results reveal the cross-environment model access and online adaptability of the Integrated-ORF, which encourages us to further study how to effectively preserve knowledge.

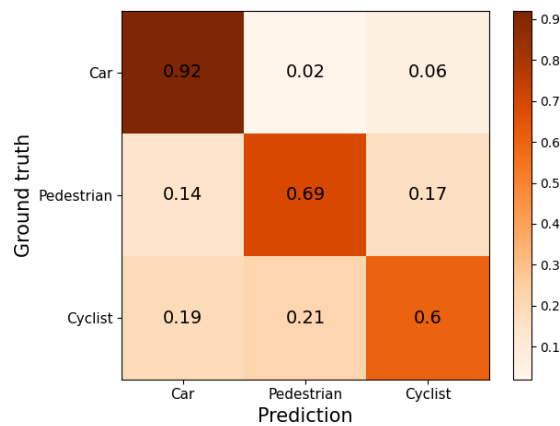


Figure 5.6: Confusion matrix representing the Integrated-ORF performance after learning on the Waymo dataset. The ordinate represents the true label, and the abscissa represents the predicted result. The darker the color, the higher the proportion of correct classifications.

## 5.5/ DISCUSSION

### 5.5.1/ ONLINE ADAPTABILITY

Online adaptability is twofold. On the one hand, OL is inherently real-time update, capable to learn new samples to update existing models in the presence of environmental

changes. On the other hand, learning for an new environment inevitably leads to catastrophic forgetting, i.e., the newly learned model is not applicable to the old situation. The performance of a well-trained model will decrease dramatically in another dataset Wang et al. (2020), which is most likely due to the fact that the parameters of the model are too close to the original domain to match the new samples and new features of the new dataset for presenting good generalization, which is particularly evident for some deep learning-based network models.

Our proposed online classifier presented excellent properties in coping with the transfer from KITTI to WAYMO dataset, with a decrease of less than 0.13 in the overall classification performance. Nevertheless, sadly there is still a priori for such adaptation. We had to adapt our segmentation and tracking methods to match the new dataset, while needing to rely on size estimation to better detect 3D objects. The consolation is that the classifier not require extensive tuning, and is able to update the existing model well by learning new samples while maintaining the original parameters. Our proposed 61-dimensional features also show good performance for object detection.

### 5.5.2/ CONVERGENCE AND STABILITY ANALYSIS

One of the challenges of online systems is how to quickly converge without manual intervention and maintain stable performance over time. Let  $u_i$  be the number of labels correctly predicted by the classifier at the  $i$ -th iteration, then the learning stability is given by:

$$stability(I) = \sum_{i=1}^I \|u_i - u_{i+1}\| \quad (5.7)$$

where in theory, according to Lyapunov stability, the iterative system will stabilize if:

$$\lim_{I \rightarrow \infty} \frac{stability(I)}{I} = 0 \quad (5.8)$$

In practice, we must face the fact that detection and tracking are imperfect, and they can go wrong at any time. Examples include the inaccurate data segmentation and sample classification in the former, and the coarse data association and target estimation in the latter. These errors may cause the learning performance of the entire framework to decline, and also cause the concept drift problem due to the fact that the model is updated as new samples are learned. Moreover, considering an ideal but edge situation in the context of long-term robot autonomy, that is the lack of ground truth in the absence of human intervention, which will also make it tricky to determine the convergence of the OL process. The solution we adopt here is to eliminate the effects of these uncertainties, including down-sampling to cope with data imbalance, dynamic parameter fine-tuning, and object size estimation.

## 5.6/ CONCLUSION

In this chapter, an efficient model training and access method for Online Learning is introduced. This method, named Online Random Forest, was originally developed in the computer vision community and was integrated under a ROS system-level framework for the first time in our research. Based on our careful investigation (see Chapter 2), it is considered to be the best among existing models that meets the Online Learning requirements for the detection of multi-class road participants in autonomous driving.

In order to integrate ORF and enable it to run efficiently, we first ROSified it, and then added support for streaming data, few-shot learning, and real-time access. In addition, in order to make model access more efficient, we studied the feasibility of replacing binary trees with octrees in ORF. Although experimental results show that compared to binary trees, octrees will increasingly save storage space as learning samples increase, but this is at the expense of model performance. In our current research, model performance is more important because it involves the evaluation of the entire Online Learning framework performance. Therefore, our research on octrees only stops at feasibility, and we report the results in this dissertation to provide a reference for the community. Future research could focus on finding a trade-off between model access and model performance.



# 6

## OVERCOMING CATASTROPHIC FORGETTING IN ONLINE LEARNING

## 6.1/ INTRODUCTION

As the field of mobile robotics continues to advance rapidly, unmanned autonomous vehicles have emerged as a promising solution within the transportation industry. The intelligence functioning of these vehicles heavily depend on their ability to effectively sense and learn objects, allowing them to swiftly identify and understand various entities in real-time, including cars, pedestrians, cyclists and other participants in the complex road situation. Over the past decade, machine learning has made remarkable progress in object detection, especially with the advent of neural network models based on deep learning which have demonstrated the capacity to surpass human capabilities (He et al., 2016; Ren et al., 2015). However, deploying machine learning methods to autonomous vehicles faces some unique challenges including expensive training, deployment and maintenance costs, domain shift, long tail problem, and so forth (Yan et al., 2023).

Compared with offline training and updating models, Online Learning (OL) is considered an effective solution (Yang et al., 2021a, 2023). An open problem in the latter is how autonomous vehicles can prevent catastrophic forgetting while continuously absorbing new knowledge. Over the past decade, many upstream methods have been proposed to address this problem in deep neural networks (Goodfellow et al., 2013; Kirkpatrick et al., 2017; Li and Hoiem, 2017). Yet these methods are inherently rooted in offline or batch training and thus, are incompatible with OL by design.

Our previous work has shown that vehicles can autonomously and rapidly learn road participant detection capabilities in a deployed environment, on-the-fly and without human intervention (Yang et al., 2021a), and with good online adaptability across environments (Yang et al., 2023). This work builds on these foundations to further investigate how to avoid the catastrophic forgetting problem in the process of in-situ learning, and still uses the detection of road participants including cars, pedestrians, and cyclists as a downstream task.

Catastrophic forgetting occurs when a model learns different tasks over multiple time slices. This problem can be crystallized into the fact that when the model generalizes to new tasks in later time slices, the performance on old tasks in early time slices drops sharply. This phenomenon creates challenges for agents that require long-term deployment, such as autonomous vehicles (Yan et al., 2020c). Research on catastrophic forgetting can be traced back to the 1990s (McCloskey and Cohen, 1989; Ratcliff, 1990). One approach is to preserve past knowledge in a way that limits changes in model weights. For instance, a memory buffer (Rolnick et al., 2019) can be employed to store data or gradient records from past training, thereby constraining the updates in the current learning process.

In continual learning, the challenge of forgetting can be defined into two main types:

class-incremental and domain-shift. The former involves adapting to new classes over time while retaining knowledge of previously learned ones. On the other hand, domain-shift focuses on adapting to shifts in the underlying data distribution. Our research primarily addresses domain shift problem, which entails adapting to new data distributions without forgetting knowledge from previous domains. Our previously proposed online learning (Yang et al., 2023) focuses on achieving high performance in short-term tasks, while long-term learning requires continual adaptation to ever-changing short-term tasks.

In situations where retaining information from previous tasks is impractical due to privacy or resource constraints, regularization-based methods (Kirkpatrick et al., 2017; Zenke et al., 2017; Schwarz et al., 2018) offer a solution by employing cleverly designed regularization losses to constrain forgetting old knowledge while learning new data. Another intuitive solution is to build a sufficiently large model and create a subset of the model for each task. This can be achieved by fixing the shared trunk and adding new branches for each new task, allowing old and new knowledge to be separated. However, this will lead to another problem of scale explosion (Li and Hoiem, 2017). Moreover, replay-based approaches are grounded on the concept of retaining or compressing the underlying data of past tasks (Lopez-Paz and Ranzato, 2017; Wang et al., 2019; Ramapuram et al., 2020). These methods combat forgetting by reintroducing stored samples during training when learning a new task, while the samples play a crucial role in joint training or loss optimization, protecting knowledge from previous tasks.

Autonomous vehicles have an essential need for agents to be able to learn on their own and continuously (Yan et al., 2023; Lesort et al., 2020), as they will be deployed into our daily lives for a long time (Vintr et al., 2022, 2019; Sun et al., 2018). In response to this need, our previous work (Yang et al., 2021a, 2023) proposed an OL framework that allows vehicles to learn the detection of road participants in-situ and on-the-fly in the environment in which they operate. However, Continual Learning (CL) across different driving scenarios brings about catastrophic forgetting problems. This motivates us to explore related prevention mechanisms within the OL framework. A work with a similar concept to the learning framework proposed in this work – to avoid forgetting in learning online – is the Lifelong Learning for Navigation (LLfN) method introduced in (Liu et al., 2021a). This method allows the robot to not forget the navigation experience of the old environment when exploring a new one. It is worth pointing out that there are essential differences between LLfN and our Long-Short-Term Online Learning (LSTOL): the former aims to learn an auxiliary planner to only help the classical planner navigate in difficult situations, while the latter treats the model to be learned and the model to be used as identical, and makes no assumptions about the usage situations. Another work similar to ours is the Expert Gate (Aljundi et al., 2017) proposed in the context of lifelong learning, which utilizes different expert networks to handle the data distribution differences across various tasks. It employs the concept of gating to effectively select experts, enabling



efficient processing of multitask situation. However, unlike LSTOL, which supports parallel online learning for multiple tasks, Expert Gate is an offline training method and requires a sequential order of learning tasks.

On the other hand, there are many works on avoiding catastrophic forgetting in computer vision (Goodfellow et al., 2013; Kirkpatrick et al., 2017; Li and Hoiem, 2017; Shmelkov et al., 2017) which is accompanied by the boom of deep learning methods. However, many of these methods cannot be straightforwardly applied to online robot learning due to their requirements for annotated data, computing resources, and training time. Our goal is thus to build an OL framework that incorporates an autonomous forgetting prevention mechanism to enable vehicles to maintain stable performance on downstream tasks during long-term operations across environments.

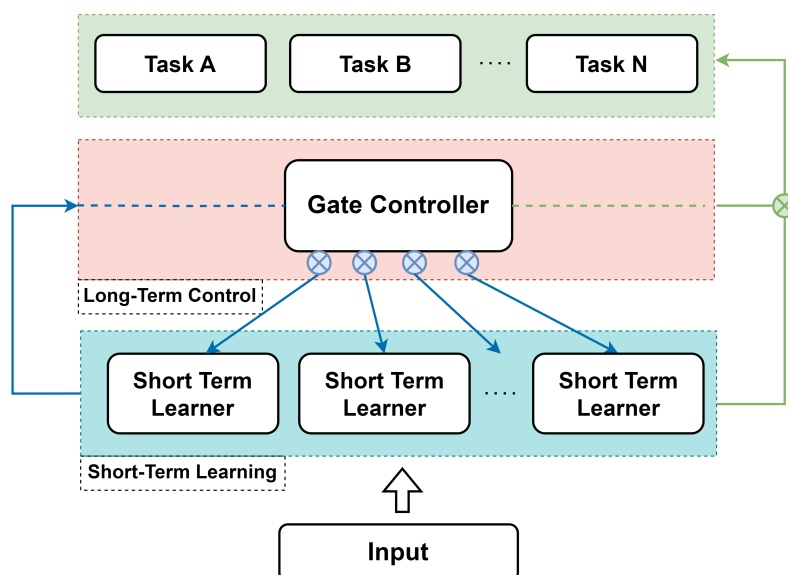


Figure 6.1: Illustration of the Long-Short-Term Online Learning (LSTOL) framework which consists of two modules: short-term learning and long-term control. Input samples are first processed by a set of short-term learners for pre-prediction. The long-term control module first collects these pre-prediction information to calculate quantitative indicators for online prediction, and then inputs it to the Gate Controller to determine the next different actions for each learner. In the learning phase (indicated by the blue line), the long-term control module will calculate the online loss of the current input sample of each learner to update the weights of the learner, which will be used in the prediction phase (indicated by the green line) to determine the object category.

Specifically, we propose an ensemble learning framework, named Long-Short-Term Online Learning (LSTOL), which consists of a set of short-term learners and a long-term control mechanism, as shown in Fig. 6.1. The former can be any model but needs to be subject to the requirements of OL, such as fast iteration without saving learning samples. The latter contains a dynamic gate controller that controls whether each existing short-term learner should be updated, kept or removed, or a new short-term learner should be

created. The design of the controller is based on primitives rather than complex reasoning, fully considering the real-time requirements of physical interaction of vehicles in the real world. It is worth mentioning that, unlike the well-known Long Short-Term Memory (LSTM) (Sherstinsky, 2020), LSTOL emphasizes the learning strategy rather than the network structure, makes no assumptions about the continuity of learning data, and allows any short-term model in design.

## 6.2/ PROBLEM FORMULATION

Environmental changes may involve temporal changes (e.g., seasons, weather) and spatial changes (e.g., locations, scenes). In the realm of online learning for object detection across diverse environments, we tackle the challenge by reframing it as a data distribution adaptation problem. The goal is to adapt an object detection model to the evolving data distribution, treating the cross-environment problem as a adaptation task.

Let  $D_t$  represent the data distribution at time step  $t$ , and  $D_{t+1}$  denote the next distribution, which signifies the ideal distribution for object detection. To formalize the adaptation process, we introduce a adaptation loss, denoted as  $L_A$ :

$$L_A = Loss(D_t, D_{t+1})$$

The objective is to minimize the distribution shift between  $D_t$  and  $D_{t+1}$ , ensuring robust object detection across varying environments.

In the proposed Long-Short-Term Online Learning (LSTOL) framework. The short-term learners, denoted as  $S_i^t$  (the state of the  $i$ -th short-term learner at time step  $t$ ), can be any model but must adhere to the requirements of Online Learning (OL), such as fast iteration without saving learning samples. The dynamic gate controller's state at time step  $t$  is represented by  $G_t$  in the long-term control mechanism, determining whether each existing short-term learner should be updated, maintain, or removed, or if a new short-term learner should be created.

To integrate these components, the adaptation mechanism adjusts the parameters of the object detection model. Let  $\theta_t$  be the model parameters at time step  $t$ . The model adaptation is performed by minimizing the sum of the object detection loss  $L_D$  and the adaptation loss:

$$\theta_{t+1} = \theta_t - \beta_t \cdot \nabla_{\theta} (L_D + \lambda \cdot L_A)$$

Here,  $\beta_t$  is the learning rate,  $\nabla_{\theta}$  represents the gradient with respect to the model parameters, and  $\lambda$  is a hyperparameter controlling the importance of adaptation in the overall learning objective.

Let  $L_t$  represent the current state of the long-term controller at time step  $t$ . The dynamic gate controller is defined as:

$$G_t = \{L_t, S_1^t, S_2^t, \dots, S_n^t\}$$

Here, the dynamic gate controller is updating based on the long-term controller's state  $L_t$  and the states of all short-term learners  $S_n^t$ .

### 6.3/ LONG-SHORT-TERM ONLINE LEARNING

As shown in Fig. 6.1, the LSTOL framework aims to cope with catastrophic forgetting by combining a short-term learning module and a long-term control module, that is, preventing the model from forgetting previously learned knowledge when learning new data. The short-term learning module consist of multiple short-term learners. Each learner can be embodied as a model such as Support Vector Machine (SVM), Random Forest (RF), Neural Network, etc., which learns from streaming data of various modalities such as images or point clouds.

The long-term control module supervises the learning of short-term learners and consists of three sub-fuction.

- **Information Collection** collects information from short-term learners, including their output confidence, accuracy and activity level on downstream tasks at the current moment. This information forms the basis for decisions about retaining existing knowledge and learning new knowledge.
- **Gate Controller** determines what actions the framework should perform including retaining, updating, or deleting existing learners, or creating new ones, based on an evaluation of the information collected and the probability calculated.
- **Weight Estimation** dynamically adjusts the corresponding weights based on the past performance of each short-term learner. If a learner's accuracy for a task is higher, its "voice" (i.e. weight) in that task increases. Learners with high prediction confidence will act as "experts" and determine the final prediction. The task with the highest confidence will be the output of the long-term control module and also the output of the entire framework.

It is worth emphasizing that the proposed LSTOL framework is learn-as-you-go, i.e., the output of the long-term control module can also be used for downstream tasks such as road participant detection.

## 6.4/ IMPLEMENTATION

An intuitive understanding of a specific implementation of the proposed LSTOL framework with point cloud-based road participant detection as a downstream task is shown in Fig. 6.2. The details are as follows.

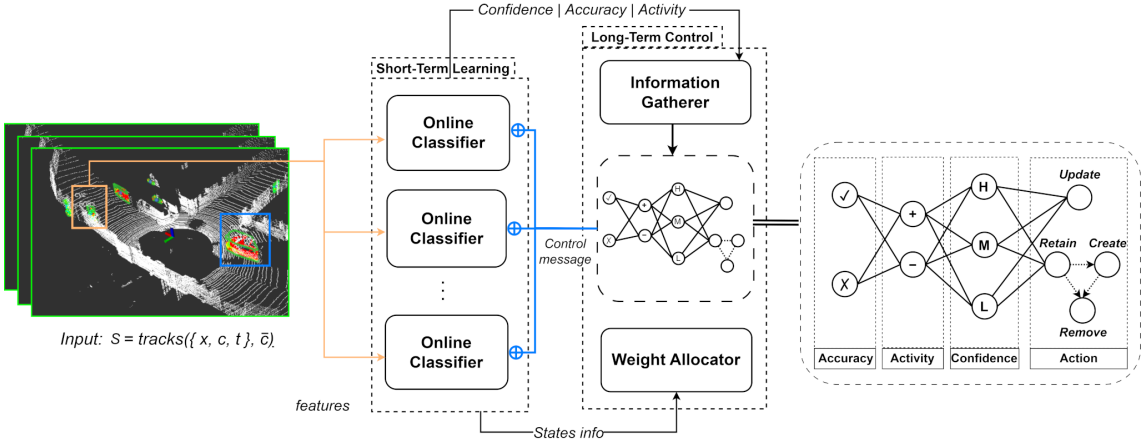


Figure 6.2: Implementation overview of the LSTOL framework. Viewed from left to right: Object samples in different point cloud frames are correlated by a multi-target tracker (Yang et al., 2023) and fed into each online classifier of the short-term learning module. The classifications are collected by the Information Gatherer in the long-term control module, which evaluates the learners based on matrices of confidence, accuracy, and activity. The evaluation information is input to the Dynamic Gate Controller, which then determines operations for each online classifier including update, retain, create, and remove. The state information of the online classifier consists of the loss for classifying the sample and is input to the Weight Allocator so that it updates the weights of the learner in different classes. These weights are also used to determine the final classification of the object.

## 6.4.1/ LEARNING SAMPLE

The learning samples of LSTOL are extracted from point clouds generated by 3D lidar mounted on autonomous vehicles, which are defined as follows:

$$S = track(\{x, c, t\}, \bar{c}) \quad (6.1)$$

where  $\{x, c, t\}$  represents a set of tracked instances (in the form of clusters) of object  $x$  at different times  $t$ .  $c$  and  $\bar{c}$  respectively represent the confidence that each instance and the entire trajectory belong to an object class.

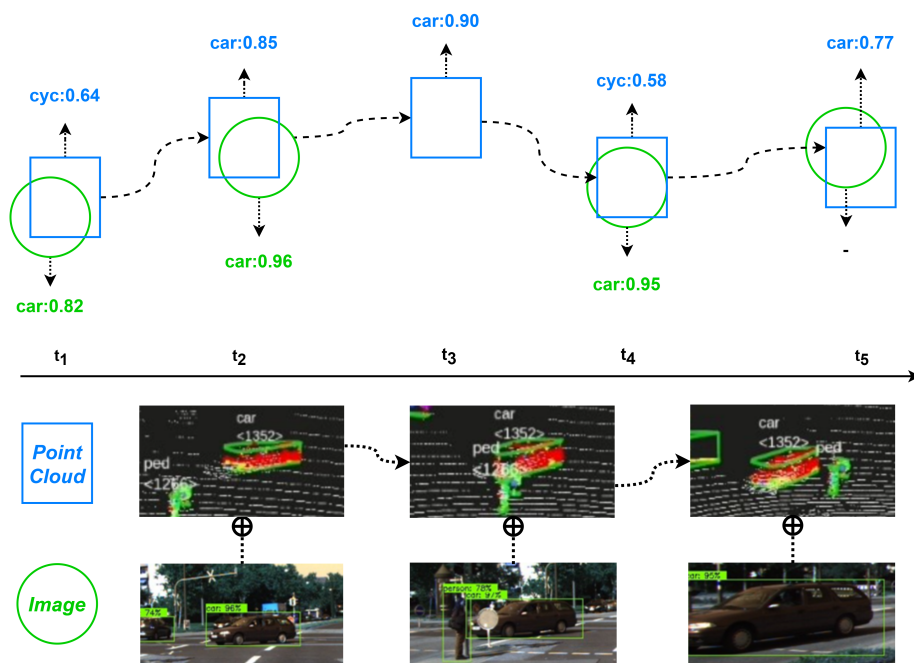


Figure 6.3: Illustrating the generation process of learning samples base on our previous work in online transfer learning (Yang et al., 2023). Rectangles denote one type of detector using point cloud which need be trained, while circles denote a well-trained detector using image.

A visualization of the generation process of learning samples is shown in Fig 6.3 which is base on our previous work in online transfer learning (Yang et al., 2023). Specifically, given an object trajectory consisting of temporal detection samples from different detectors, at any given time  $t$ , the object can be detected by one or more detectors (e.g., point cloud and image), with corresponding confidence levels indicating the likelihood of the object belonging to a particular category (illustrated in Fig with categories and probabilities). In principle, all learning samples along the trajectory should belong to the same category, and this label is determined by fusing the confidences of all samples along the trajectory. For example, despite occasional misclassifications by the point cloud-based detector, such as incorrectly categorizing a car as a cyclist (due to the performance that needs to be improved in the early stages of online learning), if the results from the pre-trained image detector and the majority of correct results from the point cloud detector indicate the object as a car, the entire trajectory will be treated as car learning samples for continual learning by the point cloud detector. Unlike the usual assumption of a fixed dataset for offline training, LSTOL processes streaming data and implements online continual learning.

### 6.4.2/ SHORT-TERM LEARNING

In the short-term learning module (denoted by  $stl$ ), each learner adopts Online Random Forest (ORF) (Saffari et al., 2009), that allows rapid multi-class model training and real-time deployment of the trained model, which is formalized as:

$$stl(x) = h \sum_{i=1}^I w_i \text{ORF}_i(x) \quad (6.2)$$

where  $w_i$  represents the weight of learner  $i$ , and  $h$  represents the fusion strategy made by the long-term control module. ORF splits a node depends on:

$$|R_j| > \alpha \quad \wedge \quad \exists s \in S : \Delta L(R_j, s) > \beta \quad (6.3)$$

where  $\alpha$  is the minimum number of samples a node needs observe before splitting,  $\beta$  is the minimum gain the split needs achieve,  $R_j$  is a decision node, and  $\Delta L(R_j, s)$  is the gain of node  $j$  with respect to test  $s$ , which is measured as:

$$\Delta L(R_j, s) = L(R_j) - \frac{|R_{jls}|}{|R_j|} L(R_{jls}) - \frac{|R_{jrs}|}{|R_j|} L(R_{jrs}) \quad (6.4)$$

where  $R_{jls}$  and  $R_{jrs}$  are the left and right partitions based on  $s$ . It can be seen that updating the ORF model only requires saving the tree structure and node information (i.e. knowledge) without the need for previous learning samples.

### 6.4.3/ LONG-TERM CONTROL

#### 6.4.3.1/ INFORMATION GATHERER

In OL, it is difficult to achieve real-time performance evaluation of models based on traditional metrics such as precision and recall due to the lack of a complete test set or ground truth for validation. Therefore, three other quantitative metrics including confidence, accuracy, and activity are used to evaluate the performance of each short-term learner, and their specific implementation is as follows.

$$Confidence_i = \max(p_i^j) \quad j = 1, \dots, J \quad (6.5)$$

where  $p_i^j$  represents the predicted probability that the object detected by learner  $i$  belongs to class  $j$ . The confidence of learner  $i$  takes the highest probability among all classes.

$$Accuracy_i = \frac{p_i^{correct}}{p_i^{total}} \quad (6.6)$$

where  $p_i^{correct}$  represents the number of correct predictions produced by learner  $i$ , and  $p_i^{total}$  represents the total number of predictions performed by learner  $i$ .

$$Activity_i = \sum_{t=1}^T update(i) \quad (6.7)$$

indicates the number of times learner  $i$  is updated in time window  $T$ .

#### 6.4.3.2/ DYNAMIC GATE CONTROLLER

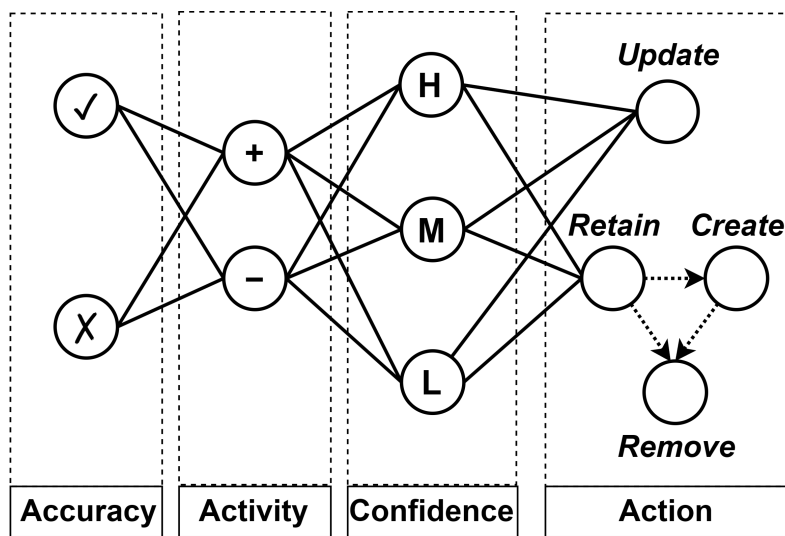


Figure 6.4: Schematic diagram of Dynamic Gate Controller

A probabilistic decision-making process is designed in this module, summarized as Algorithm 3, which implements an appropriate operation based on the three metrics provided by the Information Gatherer module (as shown in Fig. 6.4). Specifically, line-3 indicates that the learner needs to have higher accuracy before updating to prevent learning wrong data. The “retain” operation represented in line-8 corresponds to two typical situations. The first is when the learner’s accuracy is low, indicating that the learner’s predictions for new samples are beyond its knowledge. The second is when the learner has both high accuracy and high confidence, which indicates that the learner is already very familiar with the input data and does not need to learn it anymore. The idea behind line-16 is to remove those learners which have low confidence, low accuracy and low activity. Removing learners is a risky operation and therefore only occurs when the maximum number of learners is reached and new learners need to be created. Removal will inevitably lead to the forgetting of some old knowledge, but we must find a compromise between the former and the unlimited number of learners. Line-26 means that if none of the existing learners have been updated, a new one will be created.

---

**Algorithm 3** Dynamic Gate Control

---

**Require:**  $N$ : maximum number of learners to be created  $Confidence_i, Accuracy_i, Activity_i$ **Ensure:** learner  $i$ , learner  $j$ , a new learner

```

1:  $updated \leftarrow 0$ 
2: for each learner  $i \in I$  do
3:    $p \leftarrow odds(1 - Confidence_i, Accuracy_i, 1 - Activity_i)$ 
4:   if  $p > 0.5$  then
5:     update (learner  $i$ )
6:      $updated \leftarrow 1$ 
7:   else
8:     retain (learner  $i$ ) // do nothing
9:   end if
10: end for
11: if  $updated = 0$  then
12:   if  $I = N$  then
13:      $p_{max} \leftarrow 0$ 
14:      $j \leftarrow \emptyset$ 
15:     for each learner  $i \in I$  do
16:        $p \leftarrow odds(1 - Confidence_i, 1 - Accuracy_i, 1 - Activity_i)$ 
17:       if  $p > 0.5$  and  $p > p_{max}$  then
18:          $j \leftarrow i$ 
19:          $p_{max} \leftarrow p$ 
20:       end if
21:     end for
22:     if  $j \neq \emptyset$  then
23:       remove (learner  $j$ )
24:     end if
25:   else
26:     create (learner)
27:      $I \leftarrow I + 1$ 
28:   end if
29: end if

```

---

## 6.4.3.3/ WEIGHT ALLOCATOR

For each learner, a Dynamic Expert Weights (DEW) table is constructed, setting a weight for each category to represent the learner's classification ability for different classes. Classes that show better performance receive higher weights, thus amplifying their impact on the overall prediction process of the system. Specifically, assume that a new set



of samples  $S$  is input at time step  $t + 1$ , where the confidence that  $s \in S$  belongs to a certain class  $k \in K$  is very high, Kronecker delta is used to measure the sample's predicted class and actual class  $k$ , denoted as  $y_s$ . Learner  $i$ 's predicted probability that sample  $s$  belongs to class  $k$  is denoted as  $p_{s,k}$ . Then the loss of sample  $s$  is calculated by log-loss:

$$L_{s,k} = y_s \log(p_{s,k}) + (1 - y_s) \log(1 - p_{s,k}) \quad (6.8)$$

Next, the current weights  $w(t)$  are updated using an Exponentially Weighted Moving Average (EWMA), designed to reward accurate predictions and penalize incorrect ones:

$$w_k(t + 1) = \lambda w_k(t) + (1 - \lambda) L_{total} \quad (6.9)$$

where

$$L_{total} = -\frac{1}{S} \left( \sum_{s=1}^S L_{s,k} \right) \quad (6.10)$$

where  $L_{s,k}$  represents the loss of sample  $s$  of class  $k$ .  $\lambda$  is determined according to the update speed of weights which is used to balance the learner's past and present accuracy judgments.

The final prediction stage uses an intuitive voting strategy known as the Hand-raised as Expert (HraE):

$$p_k = \sum_{i, (p_{s,k} > \theta_c)}^I (p_{s,k} \cdot w_{i,k}) \quad (6.11)$$

where  $w_{i,k}$  is the weight of learner  $i$  for class  $k$ ,  $\theta_c$  represents the minimum weight required by the learner to predict, which is set to 0.5 in our experiments. Essentially, this strategy prioritizes learners' predictions based on their degree of influence in the final prediction. This allows the long-term control module to pay more attention to the predictions of modules with higher weights.

## 6.5/ EXPERIMENTAL RESULTS

### 6.5.1/ EXPERIMENTAL SETUPS

Our experiments aim to evaluate whether the proposed LSTOL can effectively prevent catastrophic forgetting when learning across environments. To this end, two very different datasets in autonomous driving including KITTI (Geiger et al., 2013) and Waymo (Sun et al., 2020) are used. Theoretically, the more different the data learned before and after, the greater the challenge in preventing catastrophic forgetting. In practice, an instance is designed to simulate an autonomous vehicle transitioning between two different environments. Specifically, the system first performs online learning on the KITTI dataset, and

then switches to the Waymo dataset for continual learning.

This operation actually causes the domain shift problem. Our previous research (Yang et al., 2023) showed that the performance of models learned on KITTI will decrease when deployed on Waymo, but by online continual learning on the latter to adapt to the new environment, the model performance will rebound.

#### 6.5.1.1/ DATASETS

Learning is initiated using randomly sampled segments from the raw data (without any annotations) of the “City”, “Residential”, and “Campus” scenes in the KITTI dataset. These three scene categories were selected because they contain a relatively large number of road participants, while the other two scenes, “Road” and “Person”, are relatively monotonous thus unsuitable for online learning of our downstream task. The system iterates the model every time it learns 100 samples and evaluates the model performance on the test set. The latter is built from randomly selected samples from the annotated training set of KITTI’s 3D object detection task, containing 5347 cars, 668 pedestrians and 271 cyclists.

In the second step, we deploy the system trained on KITTI to the Waymo dataset. We randomly selected 15 segments from the latter, for a total of 2970 images and 2970 lidar scans. These segments are dominated by daytime and clear weather conditions, and the driving scenes include cars, pedestrians and cyclists, corresponding to the three classes learned by the system on the KITTI dataset. We deliberately avoid adverse weather conditions (e.g. foggy days (Yang et al., 2020, 2021b)) and scenes with poor lighting conditions (e.g. evenings (Sun et al., 2021; Liu et al., 2021b)) to ensure a fair comparison of model performance on the two datasets. The system continues learning online on these segments. It iterates the model every 100 learned samples and evaluates its performance on the KITTI test set.

Finally, the system returns to KITTI to evaluate whether its learning on Waymo resulted in catastrophic forgetting of knowledge previously learned on KITTI.

#### 6.5.1.2/ COMPARISON MODELS

We compare our proposed system with the following methods: 1) PointNet-STD: Each dataset is independently trained based on the baseline PointNet (Qi et al., 2017a). After training on the KITTI dataset, the model parameters are further trained on the Waymo dataset to update new parameters; 2) PointNet-MIX: The baseline PointNet (Qi et al., 2017a) is jointly trained on the mixed data from the KITTI and Waymo datasets. The resulting model is directly evaluated on the KITTI validation set. 3) Expert gate (Aljundi

et al., 2017): To better suit our downstream tasks while staying true to the original algorithm, we’ve switched the neural network-based Expert Gate to an Online Expert Gate (OEG) using ORF. Still, the core idea remains unchanged: selecting the most relevant expert to handle new data based on task relevance through comparison. 4) Dynamic Expandable Network (DEN: The new implementation called 3D-DEN) (Jain and Kasaei, 2021) can perform point cloud object classification tasks end-to-end and has the capability to dynamically expand the network.

### 6.5.1.3/ IMPLEMENTATION DETAILS

We simplified the Autoencoder Gate within the Expert Gate. Instead of comparing task relevance based on validation sets of incoming data, as done in the original algorithm, we directly test the predictions of new data against the ground truth to determine task relevance. Once we find the expert with the highest task relevance, if the relevance exceeds a threshold (set at 0.85), we conduct online training on the existing expert (corresponding to LwF). Otherwise, we build a new expert to train the new data (corresponding to fine-tuning).

### 6.5.2/ EVALUATION ACROSS DATASETS

Table 6.1: Evaluation of Accuracy on KITTI Dataset before and after trained on Waymo Dataset

Method	Only KITTI		
	Car(%)	Ped(%)	Cyc(%)
PointNet-STD (Qi et al., 2017a)	<b>99.17</b>	<b>83.51</b>	<b>77.25</b>
PointNet-MIX (Qi et al., 2017a)	95.63	78.28	67.38
Expert Gate (Aljundi et al., 2017)	96.60	78.14	74.54
3D-DEN (Jain and Kasaei, 2021)	95.43	75.24	69.45
<b>Ours</b>	97.31	78.74	75.28

Method	+ Waymo					
	Car(%)		Ped(%)		Cyc(%)	
PointNet-STD (Qi et al., 2017a)	93.45	-5.72	73.79	-9.72	61.80	-15.45
PointNet-MIX (Qi et al., 2017a)	<b>95.63</b>	-	<b>78.28</b>	-	67.38	-
Expert Gate (Aljundi et al., 2017)	93.40	<b>-2.97</b>	71.26	-6.88	63.10	-11.44
3D-DEN (Jain and Kasaei, 2021)]	91.90	-3.53	69.47	-5.77	59.33	-10.12
<b>Ours</b>	93.90	-3.41	73.50	<b>-5.24</b>	<b>68.27</b>	<b>-7.01</b>

The principle of across dataset evaluation is as follows: firstly, the model will be trained on the KITTI training set and then tested on the KITTI validation set. Secondly, the model is further trained on the Waymo dataset, and the updated model will be tested on the same KITTI validation set used in the first step.

Table 6.1 presents the comparison results between four comparison models and our approach. It's worth noting that PointNet is an offline model ( $learningrate = 0.01, epochs = 20$ ). And despite the potential for Expert Gate and 3D-DEN to be used for continual and lifelong learning, they are essentially still based on offline training methods. Moreover, due to the sequential nature of their learning tasks (i.e., learning the classification tasks of cars, pedestrians, and cyclists one by one) – different from LSTOL, which supports multi-task parallel learning – it forces us to manually sort the input samples to fit its demands. In practice, however, parallelly learning multiple tasks is more beneficial for autonomous vehicles.

PointNet-based methods demonstrate superior performance under the same distribution. However, the drawbacks of this advantage are also apparent, as there is a significant decrease in the classification performance of all three categories in the evaluation of the second step. Although PointNet on mixed datasets maintains consistent performance, it's understood that this advantage may not persist as tasks and data increase, and training costs will also increase substantially. Additionally, Expert Gate and 3D-DEN show competitive results in avoiding forgetting, but our proposed LSTOL maintains an overall advantage in performance after training on the Waymo dataset, achieving the best performance. It also demonstrates the most balanced degree of performance decrease, especially in pedestrians and cyclists.

## 6.5.3/ LSTOL vs. EOTL

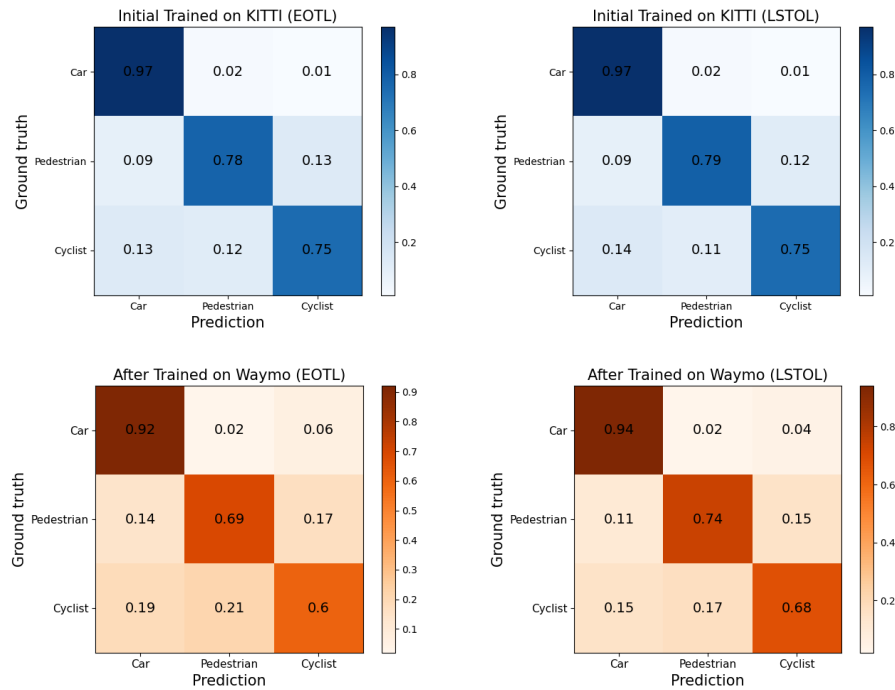


Figure 6.5: Confusion matrices for performance comparison of EOTL and LSTOL. For each matrix, the ordinate represents the true label, while the abscissa represents the predicted result. The darker the color, the higher the proportion of correct classifications.

The upper part (indicated in blue) of Fig. 6.5 shows the performance evaluation of two different models (The left matrix represents our previous method that without a forgetting prevention function, the right one represents the LSTOL proposed in this paper.) after ten learning iterations in KITTI. The lower part (indicated in orange) of Fig. 6.5 shows the performance after another ten iterations on Waymo.

From the results shown in Fig. 6.5, we can gain insight that compared with our previous online learning framework (called EOTL) that did not include a catastrophic forgetting prevention mechanism, the effect of LSTOL is obvious. Specifically, the classification performance of cars, pedestrians, and cyclists moderated from a decrease of 0.05, 0.09, and 0.15 to a decrease of 0.03, 0.05, and 0.07, respectively. Moreover, LSTOL significantly improves the ability to distinguish cars and cyclists from pedestrians.

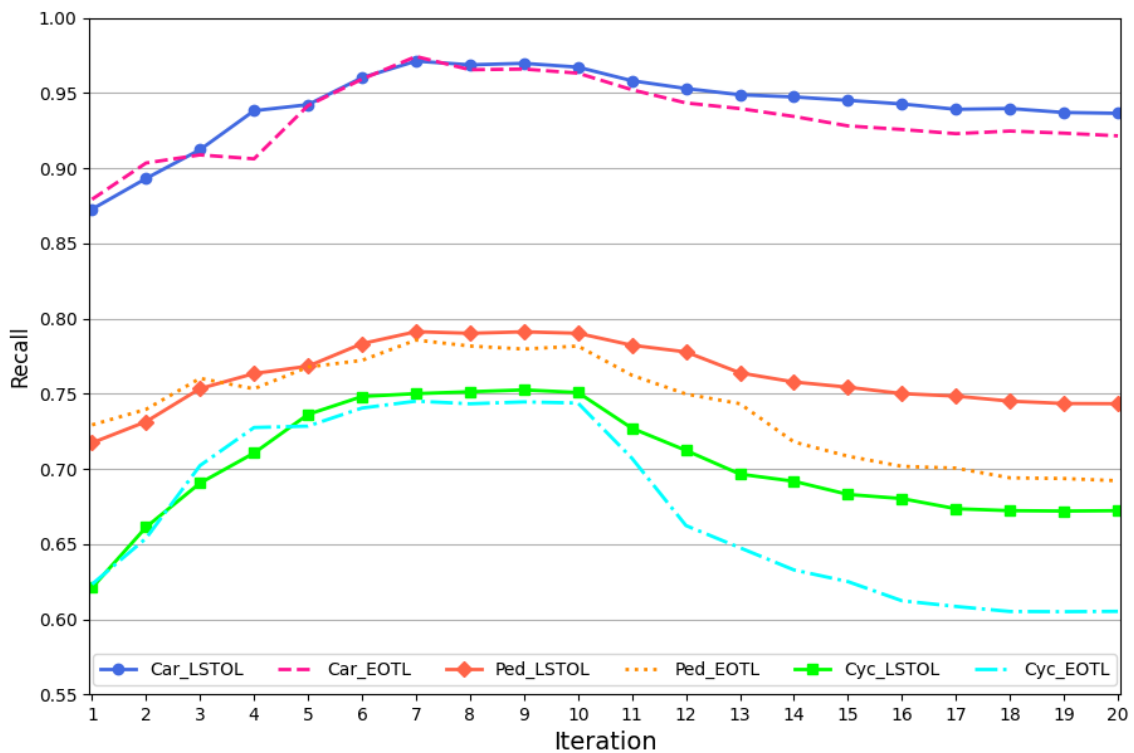


Figure 6.6: Recall curves for cars, pedestrians, and cyclists during the system transfer from KITTI to Waymo.

Furthermore, the performance of the online learned models after each iteration is visualized in Fig. 6.6, where the evaluation metric used is “recall”. In multi-class classification tasks, recall is a metric that evaluates a model’s ability to correctly identify all examples of each class. It is calculated as the ratio of the number of true positive predictions to the total number of actual positive examples in the class. Fig. 6.6 shows the results of the entire 20 iterations of moving the learning system from KITTI to Waymo. It can be seen that the anti-forgetting mechanism of LSTOL has a clear role in mitigating the performance degradation of the model caused by environmental changes, especially for pedestrians and cyclists, two types of road participants that are more difficult to detect than cars. The reason is attributed to the fact that LSTOL uses ensemble learning (a set of learners), which allows examples of each class to be better preserved independently. This is different from the single powerful learner we designed previously in EOTL, whose performance is often dominated by object classes (such as cars) that are easy to detect and have a larger number of learning examples than other classes. It can also be seen that during the learning process on the KITTI dataset, LSTOL shows better stability than EOTL. This reveals that the former can be used not only for cross-environment deployment of autonomous vehicles, but may also be suitable for long-term deployment in changing environments (Yan et al., 2020c).

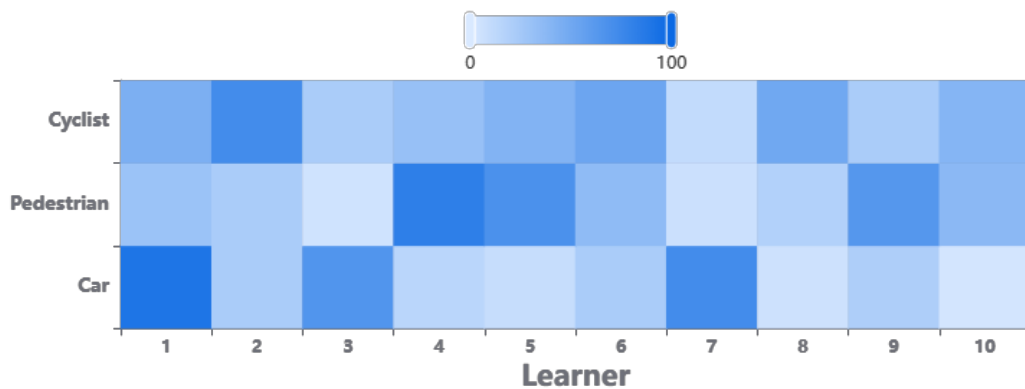


Figure 6.7: Heatmap of “expert” opinions of each learner. Each column represents the distribution of high-confidence predictions for one learner across the test samples, while each row represents high-confidence results produced by different learners for the same class.

#### 6.5.4/ SHORT-TERM LEARNER ASSESSMENT

At the micro level, we care about the performance of each short-term learner. The LSTOL model in the experiments reported in the previous section contains 10 short-term learners. 10 is an empirical number that allows both the real-time performance of the model iteration and the performance of the iterated model to be ensured. Each learner pairs an ORF, and their parameters are consistent, i.e.,  $trees = 100, depth = 50, epochs = 20, split\_threshold = 50$ . After 20 learning iterations, we evaluate each learner on KITTI’s test set, and the results are shown in Fig. 6.7.

Specifically, each test sample was predicted by the 10 learners. After filtering by the HRE mechanism, some learners’ predictions are considered as expert opinions in the final determination of object classes. The darker the color of the square in the heatmap, the higher the frequency of participating in the final prediction of that class, and vice versa. For example, Learner 1 was more involved in the classification of cars, but also contributed to the classification of pedestrians and cyclists.

It can be seen from Fig. 6.7 that each learner has a different focus on the knowledge they learn. The predictions of the first few learners show a dispersed distribution. As more new samples, such as previously unseen pedestrians and cyclists, appear in the scene, the learning goals of the learners begin to differentiate, e.g., the 2nd and 4th learners begin to focus more on pedestrians and cyclists. Starting with the 7th learner, we can see that this is the point where the dataset changes, as the learner starts learning more from new examples of cars and subsequently new pedestrians and cyclists. It can also be seen that the learner trained in Waymo provides lower confidence predictions for samples in the KITTI test set than the learner trained in KITTI.

## 6.6/ CONCLUSION

In this chapter, we introduced an Online Learning framework that incorporates mechanisms to prevent catastrophic forgetting, namely Long-Short-Term Online Learning (LSTOL). This framework aims to enable autonomous vehicles to maintain a stable performance during long-term operation in our daily lives, especially across environments. The effectiveness of the proposed framework in avoiding forgetting is demonstrated through cross datasets, including KITTI and Waymo, on downstream task of 3D detection of road participants including cars, pedestrians and cyclists.

The contributions of this work are twofold.

- A novel framework to prevent catastrophic forgetting in Online Learning is proposed. It is designed not to make assumptions about the type of short-term model and pursues a simple yet efficient strategy for long-term learning control.
- Taking road participant detection in autonomous driving as a downstream task, two very different datasets including KITTI and Waymo are used to conduct Online Continual Learning experiments, and results show that the proposed framework enables the vehicle to learn new knowledge while avoiding catastrophic forgetting.







## CONCLUSION



## GENERAL CONCLUSION

### 7.1/ CONCLUSIONS

Over the past few decades, autonomous driving has made rapid advancements, with machine perception being a pivotal foundational challenge, including crucial endeavor in detection and tracking of road participants, such as vehicles, pedestrians, and cyclists. Even though vision-based object detection has achieved significant progress in 2D and 3D detection, particularly with the emergence of deep learning based end-to-end models in recent years, there are still some crucial considerations. 1) Non-visual sensors such as 3D LiDAR have proven to have unparalleled advantages in positioning accuracy for 3D object detection, especially their insensitivity to lighting conditions. 2) The lack of interpretability in deep learning models, and their performance heavily relies on the training data. Especially when the scene or environment changes, higher training costs are required to obtain an acceptable generalization performance. Certainly, there is no perfect sensor, and the challenges posed by point cloud data that are difficult to interpret, along with the high cost of manual annotation, are primary issues in the utilize of 3D LiDAR. Moreover, in the context of autonomous driving in an open-world setting, it is imperative to address adaptability challenges brought about by variations in both time and space.

This dissertation endeavors to explore and address these challenges from three main perspectives, aligning with the three questions posed in the introduction regarding: 1) Generation of Samples, 2) Preservation of Knowledge, and 3) Avoidance of Catastrophic Forgetting. To this end, we introduce the concept of Online Continual Learning (OCL) and propose an general framework that encompasses detection, tracking, learning, and control. This framework enables models to update in real-time with new data, preserve knowledge rather than raw data, and effectively mitigate the performance degradation caused by catastrophic forgetting. In fact, Online Learning (OL) and Continual Learning (CL) are not independent concepts. On the contrary, we believe that the transition from online learning to continual learning is a consistent logical process. The framework we propose in this dissertation seeks to bridge these concepts, forming a online continual

approach to handle the dynamic and evolving nature of autonomous driving.

### **Generation of Samples**

In the face of sparse point clouds generated by 3D LiDAR and labor-intensive manual annotation, we leverage the advantages of multi-sensor and employ an efficient online transfer learning framework. This framework effectively transfers mature image-based detection capabilities to 3D LiDAR-based detectors. We construct a pipeline between such off-the-shelf detector and learn-by-use detector using a multi-target tracker to achieve knowledge transfer.

An innovative aspect is the online "learn-by-use" process, achieved through closed-loop detection, which means the output of the learned detector could be its own input after continuously iterating, and will optimizing the overall detection performance. That is, when knowledge flows over time from off-the-shelf detector to learn-by-use detector via supervised learning, learn-by-use detectors also engage in continuous self-supervised learning process. We believe that this transfer learning framework can serve as a basis for mutual learning and enhance system redundancy. Additionally, a novel information fusion strategy is proposed which combines spatio-temporal correlations, including concurrent detections of the same object in both image and point cloud spaces and temporal detections of the same object in the point cloud space. Experiments on the KITTI benchmark demonstrate the effectiveness of the proposed information fusion method for both forward (image to point cloud) and reverse (point cloud to image) transfers.

### **Preservation of Knowledge**

We introduce Online Learning to address the challenge of knowledge preservation without retaining training data, which enables autonomous systems to adapt to evolving environments. We establish three general criteria for Online Learning models: 1) rapid training, 2) immediate deployment, and 3) real-time updates, as well as a specific criterion for autonomous driving: 4) interpretability. The problem of efficient Online Learning model training and access are formulated. Innovatively, we incorporate an improved Online Random Forest (ORF) model into our online transfer learning framework, enabling the agent to quickly train models with limited computational resources and immediate deployment. Meanwhile, considering that our input data usually consists of trajectories of objects, the original Online Random Forest has been modified to handle few-shot learning. The model's initial parameters are dynamically shared throughout the training process, allowing the model to effectively address the unknown data distribution. Additionally, our exploration of the Online Random Forest tree structure demonstrates that individual extreme trees ensure the independence of each tree's training process, enhancing their ability to capture complex patterns and subtle variations in the data. Furthermore, by implementing octrees instead of binary trees to improving the storage structure, we enhance storage efficiency and accelerate model access.

### **Avoidance of Catastrophic Forgetting**

To address the inevitable forgetting problem in online learning frameworks during the long-term deploy, we propose a framework called Long-Short-Term Online Learning (LSTOL), which includes a mechanism for preventing catastrophic forgetting. LSTOL consists of a set of short-term learners and a long-term controller. The short-term learners, based on ensemble learning, aim to achieve rapid learning iterations. While the long-term controller includes a simple yet effective probabilistic decision mechanism that combines four control primitives to ensure effective knowledge maintenance. A novel feature of the proposed LSTOL framework is its ability to prevent forgetting during autonomous learning through a straightforward and effective long-term learning control strategy. Additionally, LSTOL makes no assumptions about the model types and data continuity for the short-term learners. The framework aims to maintain stable performance for autonomous vehicles, particularly in long-term, cross-environment operations in open-end world. Through cross-dataset evaluations from KITTI to Waymo, for the downstream task of 3D detection of road participants, we demonstrate the effectiveness of the proposed framework in avoiding forgetting.

## 7.2/ FUTURE RESEARCH DIRECTIONS

### **Efficient Model Training, Deployment, and Maintenance**

Future work will encompass further improving the quality of online learning samples to enhance classifier performance, resulting in improved detection and tracking. Furthermore, the choice of datasets is crucial. Exploring the use of datasets be designed to mimic real-world scenarios that accurately capture the dynamic changes in the environment over time and space, which more suitable for assessing in long-term learning performance. Additionally, it is exciting to optimizing the deployment and maintenance of autonomous vehicles equipped with our own low-power computing units.

### **Multi-sensor fusion perception**

Further research could be improve sensor fusion technology, especially the integration of different sensor combinations, not only 3D LiDAR and cameras, but also the integration of radar, thermal, depth and other sensors. Different sensor fusion can enhance object detection and tracking in challenging scenes, providing a more comprehensive understanding of the environment. Furthermore, multi-sensor-based redundancy mechanisms and strategies can be explored to ensure that autonomous vehicles can continue to operate safely even in the event of malfunctions or unexpected events.

### **Long-term Continual Learning in Dynamic Environments**

Future research should explore more deeply the strategies of how autonomous vehicles adapts to dynamic changes in the environment. This adaptation includes not only adjusting to variations in the physical environment, such as weather and lighting conditions but also accommodating changes in traffic rules, traffic signal, and other real-world factors that can impact autonomous driving systems. Meanwhile, future work will investigate mechanisms to dynamically control the number of short-term learners to eliminate the a prior need for this number, allowing autonomous systems to evolve and adapt in real-time. Furthermore, it's important to test the generalizability of upstream methods to various downstream tasks in dynamic environments, such as human-aware navigation Vintr et al. (2022); Okunevich et al. (2023).

# BIBLIOGRAPHY

- [Aljundi et al. 2017] ALJUNDI, Rahaf ; CHAKRAVARTY, Punarjay ; TUYTELAARS, Tinne: **“Expert gate: Lifelong learning with a network of experts”**. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pages 3366–3375
- [Bordes et al. 2005] BORDES, Antoine ; ERTEKIN, Seyda ; WESTON, Jason ; BOTTON, Léon ; CRISTIANINI, Nello: **“Fast kernel classifiers with online and active learning.”**. In *Journal of Machine Learning Research* 6 (2005), number 9
- [Bottou 2010] BOTTOU, Léon: **“Large-scale machine learning with stochastic gradient descent”**. In *Proceedings of COMPSTAT'2010: 19th International Conference on Computational Statistics Paris France, August 22-27, 2010 Keynote, Invited and Contributed Papers* Springer (event), 2010, pages 177–186
- [Broughton et al. 2020] BROUGHTON, George ; MAJER, Filip ; ROUČEK, Tomáš ; RUICHEK, Yassine ; YAN, Zhi ; KRAJNÍK, Tomáš: **“Learning to see through the haze: Multi-sensor learning-fusion System for Vulnerable Traffic Participant Detection in Fog”**. In *Robotics and Autonomous Systems* (2020), pages 103687
- [Cauwenberghs and Poggio 2000] CAUWENBERGHS, Gert ; POGGIO, Tomaso: **“Incremental and decremental support vector machine learning”**. In *Advances in neural information processing systems* 13 (2000)
- [Chaudhry et al. 2018] CHAUDHRY, Arslan ; RANZATO, Marc'Aurelio ; ROHRBACH, Marcus ; ELHOSEINY, Mohamed: **“Efficient lifelong learning with a-gem”**. In *arXiv preprint arXiv:1812.00420* (2018)
- [Chen et al. 2016] CHEN, Xiaozhi ; KUNDU, Kaustav ; ZHANG, Ziyu ; MA, Huimin ; FIDLER, Sanja ; URTASUN, Raquel: **“Monocular 3d object detection for autonomous driving”**. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pages 2147–2156
- [Chen et al. 2017] CHEN, Xiaozhi ; MA, Huimin ; WAN, Ji ; LI, Bo ; XIA, Tian: **“Multi-view 3d object detection network for autonomous driving”**. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pages 1907–1915
- [Chen et al. 2022] CHEN, Xieyuanli ; MERSCH, Benedikt ; NUNES, Lucas ; MARCUZZI, Rodrigo ; VIZZO, Ignacio ; BEHLEY, Jens ; STACHNISS, Cyrill: **“Automatic Labeling to**



- Generate Training Data for Online LiDAR-Based Moving Object Segmentation**". In *IEEE Robotics and Automation Letters* 7 (2022), number 3, pages 6107–6114
- [Dequaire et al. 2018] DEQUAIRE, Julie ; ONDRŮŠKA, Peter ; RAO, Dushyant ; WANG, Dominic ; POSNER, Ingmar: **"Deep tracking in the wild: End-to-end tracking using recurrent neural networks"**. In *The International Journal of Robotics Research* 37 (2018), number 4-5, pages 492–512
- [Dhar et al. 2019] DHAR, Prithviraj ; SINGH, Rajat V. ; PENG, Kuan-Chuan ; WU, Ziyang ; CHELLAPPA, Rama: **"Learning without memorizing"**. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pages 5138–5146
- [Ertekin et al. 2010] ERTEKIN, Seyda ; BOTTOU, Leon ; GILES, C L.: **"Nonconvex online support vector machines"**. In *IEEE Transactions on Pattern Analysis and Machine Intelligence* 33 (2010), number 2, pages 368–381
- [Freund et al. 1999] FREUND, Yoav ; SCHAPIRE, Robert ; ABE, Naoki: **"A short introduction to boosting"**. In *Journal-Japanese Society For Artificial Intelligence* 14 (1999), number 771-780, pages 1612
- [Geiger et al. 2013] GEIGER, Andreas ; LENZ, Philip ; STILLER, Christoph ; URTASUN, Raquel: **"Vision meets robotics: The kitti dataset"**. In *The International Journal of Robotics Research* 32 (2013), number 11, pages 1231–1237
- [Geiger et al. 2012] GEIGER, Andreas ; LENZ, Philip ; URTASUN, Raquel: **"Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite"**. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012
- [Geurts et al. 2006] GEURTS, Pierre ; ERNST, Damien ; WEHENKEL, Louis: **"Extremely randomized trees"**. In *Machine learning* 63 (2006), number 1, pages 3–42
- [González et al. 2016] GONZÁLEZ, Alejandro ; VÁZQUEZ, David ; LÓPEZ, Antonio M. ; AMORES, Jaume: **"On-board object detection: Multicue, multimodal, and multi-view random forest of local experts"**. In *IEEE transactions on cybernetics* 47 (2016), number 11, pages 3980–3990
- [Goodfellow et al. 2013] GOODFELLOW, Ian J. ; MIRZA, Mehdi ; XIAO, Da ; COURVILLE, Aaron ; BENGIO, Yoshua: **"An empirical investigation of catastrophic forgetting in gradient-based neural networks"**. In *arXiv preprint arXiv:1312.6211* (2013)
- [Hadsell et al. 2020] HADSELL, Raia ; RAO, Dushyant ; RUSU, Andrei A. ; PASCANU, Razvan: **"Embracing change: Continual learning in deep neural networks"**. In *Trends in cognitive sciences* 24 (2020), number 12, pages 1028–1040

- [He et al. 2016] HE, Kaiming ; ZHANG, Xiangyu ; REN, Shaoqing ; SUN, Jian: **“Deep residual learning for image recognition”**. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pages 770–778
- [Hoi et al. 2021] HOI, Steven C. ; SAHOO, Doyen ; LU, Jing ; ZHAO, Peilin: **“Online learning: A comprehensive survey”**. In *Neurocomputing* 459 (2021), pages 249–289
- [Hornung et al. 2013] HORNUNG, Armin ; WURM, Kai M. ; BENNEWITZ, Maren ; STACHNISS, Cyrill ; BURGARD, Wolfram: **“OctoMap: An efficient probabilistic 3D mapping framework based on octrees”**. In *Autonomous robots* 34 (2013), pages 189–206
- [Jain and Kasaei 2021] JAIN, Sudhakaran ; KASAEI, Hamidreza: **“3D\_DEN: Open-ended 3D object recognition using dynamically expandable networks”**. In *IEEE Transactions on Cognitive and Developmental Systems* (2021)
- [Kato et al. 2018] KATO, Shinpei ; TOKUNAGA, Shota ; MARUYAMA, Yuya ; MAEDA, Seiya ; HIRABAYASHI, Manato ; KITSUKAWA, Yuki ; MONRROY, Abraham ; ANDO, Tomohito ; FUJII, Yusuke ; AZUMI, Takuya: **“Autoware on board: enabling autonomous vehicles with embedded systems”**. In *Proceedings of the 9th ACM/IEEE International Conference on Cyber-Physical Systems (ICCPs)*, 2018, pages 287–296
- [Kidono et al. 2011] KIDONO, Kiyosumi ; MIYASAKA, Takeo ; WATANABE, Akihiro ; NAITO, Takashi ; MIURA, Jun: **“Pedestrian recognition using high-definition LIDAR”**. In *Proceedings of the 2011 IEEE Intelligent Vehicles Symposium (IV)*, 2011, pages 405–410
- [Kingma and Welling 2013] KINGMA, Diederik P. ; WELLING, Max: **“Auto-encoding variational bayes”**. In *arXiv preprint arXiv:1312.6114* (2013)
- [Kirkpatrick et al. 2017] KIRKPATRICK, James ; PASCANU, Razvan ; RABINOWITZ, Neil ; VENESS, Joel ; DESJARDINS, Guillaume ; RUSU, Andrei A. ; MILAN, Kieran ; QUAN, John ; RAMALHO, Tiago ; GRABSKA-BARWINSKA, Agnieszka ; OTHERS: **“Overcoming catastrophic forgetting in neural networks”**. In *Proceedings of the national academy of sciences* 114 (2017), number 13, pages 3521–3526
- [Kolouri et al. 2019] KOLOURI, Soheil ; KETZ, Nicholas ; ZOU, Xinyun ; KRICHMAR, Jeffrey ; PILLY, Praveen: **“Attention-based structural-plasticity”**. In *arXiv preprint arXiv:1903.06070* (2019)
- [Krajník et al. 2017] KRAJNÍK, Tomáš ; FENTANES, Jaime P. ; SANTOS, Joao M. ; DUCKETT, Tom: **“Fremen: Frequency map enhancement for long-term mobile robot autonomy in changing environments”**. In *IEEE Transactions on Robotics* 33 (2017), number 4, pages 964–977

- [**Ku et al. 2018**] KU, Jason ; MOZIFIAN, Melissa ; LEE, Jungwook ; HARAKEH, Ali ; WASLANDER, Steven L.: **“Joint 3d proposal generation and object detection from view aggregation”**. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* IEEE (event), 2018, pages 1–8
- [**Kuznetsova et al. 2015**] KUZNETSOVA, Alina ; JU HWANG, Sung ; ROSENHAHN, Bodo ; SIGAL, Leonid: **“Expanding object detector’s horizon: Incremental learning framework for object detection in videos”**. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pages 28–36
- [**Lesort et al. 2020**] LESORT, Timothée ; LOMONACO, Vincenzo ; STOIAN, Andrei ; MALTONI, Davide ; FILLIAT, David ; DÍAZ-RODRÍGUEZ, Natalia: **“Continual learning for robotics: Definition, framework, learning strategies, opportunities and challenges”**. In *Information Fusion* 58 (2020), pages 52–68
- [**Li and Hoiem 2017**] LI, Zhizhong ; HOIEM, Derek: **“Learning without forgetting”**. In *IEEE transactions on pattern analysis and machine intelligence* 40 (2017), number 12, pages 2935–2947
- [**Liang et al. 2006**] LIANG, Nan-Ying ; HUANG, Guang-Bin ; SARATCHANDRAN, Paramasivan ; SUNDARARAJAN, Narasimhan: **“A fast and accurate online sequential learning algorithm for feedforward networks”**. In *IEEE Transactions on neural networks* 17 (2006), number 6, pages 1411–1423
- [**Liu et al. 2021a**] LIU, Bo ; XIAO, Xuesu ; STONE, Peter: **“A lifelong learning approach to mobile robot navigation”**. In *IEEE Robotics and Automation Letters* 6 (2021), number 2, pages 1090–1096
- [**Liu et al. 2021b**] LIU, Linrunjia ; CAPPELLE, Cindy ; RUICHEK, Yassine: **“Day and Night Place Recognition Based on Low-quality Night-time Images”**. In *Proceedings of the 2020 IEEE International Conference on Intelligent Transportation Systems (ITSC)*, 2021, pages 1–6
- [**Loh 2011**] LOH, Wei-Yin: **“Classification and regression trees”**. In *Wiley interdisciplinary reviews: data mining and knowledge discovery* 1 (2011), number 1, pages 14–23
- [**Lomonaco and Maltoni 2017**] LOMONACO, Vincenzo ; MALTONI, Davide: **“Core50: a new dataset and benchmark for continuous object recognition”**. In *Conference on robot learning* PMLR (event), 2017, pages 17–26
- [**Lopez-Paz and Ranzato 2017**] LOPEZ-PAZ, David ; RANZATO, Marc’Aurelio: **“Gradient episodic memory for continual learning”**. In *Advances in neural information processing systems* 30 (2017)

- [Ma et al. 2018] MA, Jiaqi ; ZHAO, Zhe ; YI, Xinyang ; CHEN, Jilin ; HONG, Lichan ; CHI, Ed H.: **“Modeling task relationships in multi-task learning with multi-gate mixture-of-experts”**. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, 2018, pages 1930–1939
- [Mai et al. 2022] MAI, Zheda ; LI, Ruiwen ; JEONG, Jihwan ; QUISPE, David ; KIM, Hyunwoo ; SANNER, Scott: **“Online continual learning in image classification: An empirical survey”**. In *Neurocomputing* 469 (2022), pages 28–51
- [Majer et al. 2019] MAJER, Filip ; YAN, Zhi ; BROUGHTON, George ; RUICHEK, Yassine ; KRAJNIK, Tomas: **“Learning to see through haze: Radar-based human detection for adverse weather conditions”**. In *Proceedings of ECMR*. Prague, Czech Republic, September 2019
- [Maltoni and Lomonaco 2019] MALTONI, Davide ; LOMONACO, Vincenzo: **“Continuous learning in single-incremental-task scenarios”**. In *Neural Networks* 116 (2019), pages 56–73
- [Masana et al. 2022] MASANA, Marc ; LIU, Xialei ; TWARDOWSKI, Bartłomiej ; MENTA, Mikel ; BAGDANOV, Andrew D. ; VAN DE WEIJER, Joost: **“Class-incremental learning: survey and performance evaluation on image classification”**. In *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45 (2022), number 5, pages 5513–5533
- [McCloskey and Cohen 1989] MCCLOSKEY, Michael ; COHEN, Neal J.: **“Catastrophic interference in connectionist networks: The sequential learning problem”**. In *Psychology of learning and motivation* Volume 24. Elsevier, 1989, pages 109–165
- [Mousavian et al. 2017] MOUSAVIAN, Arsalan ; ANGUELOV, Dragomir ; FLYNN, John ; KOSECKA, Jana: **“3d bounding box estimation using deep learning and geometry”**. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pages 7074–7082
- [Navarro-Serment et al. 2009] NAVARRO-SERMENT, Luis E. ; MERTZ, Christoph ; HEBERT, Martial: **“Pedestrian Detection and Tracking Using Three-Dimensional LADAR Data”**. In *Proceedings of the 7th Conference on Field and Service Robotics (FSR)*, 2009, pages 103–112
- [Okunevich et al. 2023] OKUNEVICH, Iaroslav ; HILAIRE, Vincent ; GALLAND, Stephane ; LAMOTTE, Olivier ; SHILOVA, Liubov ; RUICHEK, Yassine ; YAN, Zhi: **“Human-centered Benchmarking for Socially-compliant Robot Navigation”**. In *Proceedings of the 2023 European Conference on Mobile Robots (ECMR)*, 2023, pages 1–7
- [Padilla et al. 2020] PADILLA, Rafael ; NETTO, Sergio L. ; DA SILVA, Eduardo A.: **“A survey on performance metrics for object-detection algorithms”**. In *2020 interna-*

*tional conference on systems, signals and image processing (IWSSIP) IEEE (event), 2020, pages 237–242*

- [Pan and Yang 2009] PAN, Sinno J. ; YANG, Qiang: **“A survey on transfer learning”**. In *IEEE Transactions on knowledge and data engineering* 22 (2009), number 10, pages 1345–1359
- [Pedregosa et al. 2011] PEDREGOSA, F. ; VAROQUAUX, G. ; GRAMFORT, A. ; MICHEL, V. ; THIRION, B. ; GRISEL, O. ; BLONDEL, M. ; PRETTENHOFER, P. ; WEISS, R. ; DUBOURG, V. ; VANDERPLAS, J. ; PASSOS, A. ; COURNAPEAU, D. ; BRUCHER, M. ; PERROT, M. ; DUCHESNAY, E.: **“Scikit-learn: Machine Learning in Python”**. In *Journal of Machine Learning Research* 12 (2011), pages 2825–2830
- [Perez-Rua et al. 2020] PEREZ-RUA, Juan-Manuel ; ZHU, Xiatian ; HOSPEDALES, Timothy M. ; XIANG, Tao: **“Incremental few-shot object detection”**. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020*, pages 13846–13855
- [Platt 1998] PLATT, John: **“Sequential minimal optimization: A fast algorithm for training support vector machines”**. (1998)
- [Polikar et al. 2001] POLIKAR, Robi ; UPDA, Lalita ; UPDA, Satish S. ; HONAVAR, Vasant: **“Learn++: An incremental learning algorithm for supervised neural networks”**. In *IEEE transactions on systems, man, and cybernetics, part C (applications and reviews)* 31 (2001), number 4, pages 497–508
- [Qi et al. 2018] QI, Charles R. ; LIU, Wei ; WU, Chenxia ; SU, Hao ; GUIBAS, Leonidas J.: **“Frustum pointnets for 3d object detection from rgb-d data”**. In *Proceedings of the IEEE conference on computer vision and pattern recognition, 2018*, pages 918–927
- [Qi et al. 2017a] QI, Charles R. ; SU, Hao ; MO, Kaichun ; GUIBAS, Leonidas J.: **“Pointnet: Deep learning on point sets for 3d classification and segmentation”**. In *Proceedings of the IEEE conference on computer vision and pattern recognition, 2017*, pages 652–660
- [Qi et al. 2017b] QI, Charles R. ; YI, Li ; SU, Hao ; GUIBAS, Leonidas J.: **“Pointnet++: Deep hierarchical feature learning on point sets in a metric space”**. In *Advances in neural information processing systems* 30 (2017)
- [Qian et al. 2022] QIAN, Rui ; LAI, Xin ; LI, Xirong: **“3D object detection for autonomous driving: a survey”**. In *Pattern Recognition* (2022), pages 108796
- [Ramapuram et al. 2020] RAMAPURAM, Jason ; GREGOROVA, Magda ; KALOUSIS, Alexandros: **“Lifelong generative modeling”**. In *Neurocomputing* 404 (2020), pages 381–400

- [Ratcliff 1990] RATCLIFF, Roger: **“Connectionist models of recognition memory: constraints imposed by learning and forgetting functions.”**. In *Psychological review* 97 (1990), number 2, pages 285
- [Redmon and Farhadi 2018] REDMON, Joseph ; FARHADI, Ali: **“Yolov3: An incremental improvement”**. In *arXiv preprint arXiv:1804.02767* (2018)
- [Ren et al. 2015] REN, Shaoqing ; HE, Kaiming ; GIRSHICK, Ross ; SUN, Jian: **“Faster r-cnn: Towards real-time object detection with region proposal networks”**. In *Advances in neural information processing systems* 28 (2015)
- [Ring 1994] RING, Mark B.: *Continual Learning in Reinforcement Environments*, The University of Texas at Austin, PhD Thesis, August 1994
- [Ritter et al. 2018] RITTER, Hippolyt ; BOTEV, Aleksandar ; BARBER, David: **“Online structured laplace approximations for overcoming catastrophic forgetting”**. In *Advances in Neural Information Processing Systems* 31 (2018)
- [Robins 1995] ROBINS, Anthony: **“Catastrophic forgetting, rehearsal and pseudorehearsal”**. In *Connection Science* 7 (1995), number 2, pages 123–146
- [Rolnick et al. 2019] ROLNICK, David ; AHUJA, Arun ; SCHWARZ, Jonathan ; LILLICRAP, Timothy ; WAYNE, Gregory: **“Experience replay for continual learning”**. In *Advances in Neural Information Processing Systems* 32 (2019)
- [Rumelhart et al. 1985] RUMELHART, David E. ; HINTON, Geoffrey E. ; WILLIAMS, Ronald J. ; OTHERS: *Learning internal representations by error propagation*. 1985
- [Rusu 2009] RUSU, Radu B.: *Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments*, Computer Science department, Technische Universitaet Muenchen, Germany, PhD Thesis, 2009
- [Ruvolo and Eaton 2013] RUVOLO, Paul ; EATON, Eric: **“ELLA: An efficient lifelong learning algorithm”**. In *International conference on machine learning* PMLR (event), 2013, pages 507–515
- [Saffari et al. 2009] SAFFARI, Amir ; LEISTNER, Christian ; SANTNER, Jakob ; GODEC, Martin ; BISCHOF, Horst: **“On-line random forests”**. In *2009 IEEE 12th international conference on computer vision workshops, ICCV workshops, 2009*, pages 1393–1400
- [Schwarz et al. 2018] SCHWARZ, Jonathan ; CZARNECKI, Wojciech ; LUKETINA, Jelena ; GRABSKA-BARWINSKA, Agnieszka ; TEH, Yee W. ; PASCANU, Razvan ; HADSELL, Raia: **“Progress & compress: A scalable framework for continual learning”**. In *International conference on machine learning* PMLR (event), 2018, pages 4528–4537

- [Shazeer et al. 2017] SHAZEER, Noam ; MIRHOSEINI, Azalia ; MAZIARZ, Krzysztof ; DAVIS, Andy ; LE, Quoc ; HINTON, Geoffrey ; DEAN, Jeff: **“Outrageously large neural networks: The sparsely-gated mixture-of-experts layer”**. In *arXiv preprint arXiv:1701.06538* (2017)
- [Sherstinsky 2020] SHERSTINSKY, Alex: **“Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network”**. In *Physica D: Nonlinear Phenomena* 404 (2020), pages 132306
- [Shmelkov et al. 2017] SHMELKOV, Konstantin ; SCHMID, Cordelia ; ALAHARI, Karteek: **“Incremental learning of object detectors without catastrophic forgetting”**. In *Proceedings of the IEEE international conference on computer vision*, 2017, pages 3400–3409
- [Sun et al. 2021] SUN, Li ; TAHER, Marwan ; WILD, Christopher ; ZHAO, Cheng ; MAJER, Filip ; YAN, Zhi ; KRAJNIK, Tomas ; PRESCOTT, Tony J. ; DUCKETT, Tom: **“Robust and long-term monocular teach-and-repeat navigation using a single-experience map”**. In *Proceedings of the 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pages 2635–2642
- [Sun et al. 2018] SUN, Li ; YAN, Zhi ; MELLADO, Sergi M. ; HANHEIDE, Marc ; DUCKETT, Tom: **“3DOF Pedestrian Trajectory Prediction Learned from Long-Term Autonomous Mobile Robot Deployment Data”**. In *Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018
- [Sun et al. 2020] SUN, Pei ; KRETZSCHMAR, Henrik ; DOTIWALLA, Xerxes ; CHOUARD, Aurelien ; PATNAIK, Vijaysai ; TSUI, Paul ; GUO, James ; ZHOU, Yin ; CHAI, Yuning ; CAINE, Benjamin ; OTHERS: **“Scalability in perception for autonomous driving: Waymo open dataset”**. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pages 2446–2454
- [Tan et al. 2020] TAN, Mingxing ; PANG, Ruoming ; LE, Quoc V.: **“Efficientdet: Scalable and efficient object detection”**. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pages 10781–10790
- [Teichman and Thrun 2011] TEICHMAN, Alex ; THRUN, Sebastian: **“Tracking-based semi-supervised learning”**. In *Proceedings of Robotics: Science and Systems*, 2011
- [Thrun 1994] THRUN, Sebastian: **“A lifelong learning perspective for mobile robot control”**. In *Proceedings of the 1994 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 1994, pages 23–30
- [Vintr et al. 2022] VINTR, Tomas ; BLAHA, Jan ; REKTORIS, Martin ; ULRICH, Jiri ; ROUCEK, Tomas ; BROUGHTON, George ; YAN, Zhi ; KRAJNIK, Tomas: **“Toward**

- benchmarking of long-term spatio-temporal maps of pedestrian flows for human-aware navigation**". In *Frontiers in Robotics and AI* 9 (2022), pages 890013
- [Vintr et al. 2019] VINTR, Tomas ; YAN, Zhi ; DUCKETT, Tom ; KRAJNIK, Tomas: **"Spatio-temporal Representation for Long-term Anticipation of Human Presence in Service Robotics"**. In *Proceedings of the 2019 IEEE International Conference on Robotics and Automation (ICRA)*, 2019, pages 2620–2626
- [Wang et al. 2019] WANG, Hong ; XIONG, Wenhan ; YU, Mo ; GUO, Xiaoxiao ; CHANG, Shiyu ; WANG, William Y.: **"Sentence embedding alignment for lifelong relation extraction"**. In *arXiv preprint arXiv:1903.02588* (2019)
- [Wang et al. 2021] WANG, Jianren ; WANG, Xin ; SHANG-GUAN, Yue ; GUPTA, Abhinav: **"Wanderlust: Online continual object detection in the real world"**. In *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pages 10829–10838
- [Wang et al. 2020] WANG, Yan ; CHEN, Xiangyu ; YOU, Yurong ; LI, Li E. ; HARIHARAN, Bharath ; CAMPBELL, Mark ; WEINBERGER, Kilian Q. ; CHAO, Wei-Lun: **"Train in germany, test in the usa: Making 3d object detectors generalize"**. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pages 11713–11723
- [Wen and Jo 2021] WEN, Li-Hua ; JO, Kang-Hyun: **"Fast and accurate 3D object detection for lidar-camera-based autonomous vehicles using one shared voxel-based backbone"**. In *IEEE Access* 9 (2021), pages 22080–22089
- [Xu and Chen 2018] XU, Bin ; CHEN, Zhenzhong: **"Multi-level fusion based 3d object detection from monocular images"**. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pages 2345–2353
- [Yan et al. 2017] YAN, Zhi ; DUCKETT, Tom ; BELLOTTO, Nicola: **"Online learning for human classification in 3d lidar-based tracking"**. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pages 864–871
- [Yan et al. 2020a] YAN, Zhi ; DUCKETT, Tom ; BELLOTTO, Nicola: **"Online learning for 3D LiDAR-based human detection: experimental analysis of point cloud clustering and classification methods"**. In *Autonomous Robots* 44 (2020), pages 147–164
- [Yan et al. 2020b] YAN, Zhi ; SCHREIBERHUBER, Simon ; HALMETSCHLAGER, Georg ; DUCKETT, Tom ; VINCZE, Markus ; BELLOTTO, Nicola: **"Robot Perception of Static and Dynamic Objects with an Autonomous Floor Scrubber"**. In *Intelligent Service Robotics* 13 (2020), number 3, pages 403–417



- [Yan et al. 2018] YAN, Zhi ; SUN, Li ; DUCKETT, Tom ; BELLOTTO, Nicola: **“Multisensor Online Transfer Learning for 3D LiDAR-based Human Detection with a Mobile Robot”**. In *In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018
- [Yan et al. 2023] YAN, Zhi ; SUN, Li ; KRAJNIK, Tomas ; DUCKETT, Tom ; BELLOTTO, Nicola: **“Towards Long-term Autonomy: A Perspective from Robot Learning”**. In *AAAI-23 Bridge Program on AI & Robotics*, 2023
- [Yan et al. 2020c] YAN, Zhi ; SUN, Li ; KRAJNIK, Tomas ; RUICHEK, Yassine: **“EU Long-term Dataset with Multiple Sensors for Autonomous Driving”**. In *Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Las Vegas, USA, October 2020
- [Yang et al. 2021a] YANG, Rui ; YAN, Zhi ; RUICHEK, Yassine ; YANG, Tao: **“Efficient Online Transfer Learning for 3D Object Classification in Autonomous Driving”**. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, 2021, pages 2950–2957
- [Yang et al. 2023] YANG, Rui ; YAN, Zhi ; YANG, Tao ; WANG, Yaonan ; RUICHEK, Yassine: **“Efficient Online Transfer Learning for Road Participants Detection in Autonomous Driving”**. In *IEEE Sensors Journal* (2023)
- [Yang et al. 2020] YANG, Tao ; LI, You ; RUICHEK, Yassine ; YAN, Zhi: **“LaNoising: A Data-driven Approach for 903nm ToF LiDAR Performance Modeling under Fog”**. In *Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pages 10084–10091
- [Yang et al. 2021b] YANG, Tao ; LI, You ; RUICHEK, Yassine ; YAN, Zhi: **“Performance modeling a near-infrared ToF LiDAR under fog: A data-driven approach”**. In *IEEE Transactions on Intelligent Transportation Systems* 23 (2021), number 8, pages 11227–11236
- [Yang et al. 2022] YANG, Tao ; LI, You ; ZHAO, Cheng ; YAO, Dexin ; CHEN, Guanyin ; SUN, Li ; KRAJNIK, Tomas ; YAN, Zhi: **“3D ToF LiDAR in Mobile Robotics: A Review”**. In *arXiv preprint arXiv:2202.11025* (2022)
- [Yoon et al. 2017] YOON, Jaehong ; YANG, Eunho ; LEE, Jeongtae ; HWANG, Sung J.: **“Lifelong learning with dynamically expandable networks”**. In *arXiv preprint arXiv:1708.01547* (2017)
- [Zenke et al. 2017] ZENKE, Friedemann ; POOLE, Ben ; GANGULI, Surya: **“Continual learning through synaptic intelligence”**. In *International conference on machine learning* PMLR (event), 2017, pages 3987–3995

- [Zermas et al. 2017] ZERMAS, Dimitris ; IZZAT, Izzat ; PAPANIKOLOPOULOS, Nikolaos: **“Fast segmentation of 3d point clouds: A paradigm on lidar data for autonomous vehicle applications”**. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pages 5067–5073



# LIST OF FIGURES

1.1	Annotated example of a point cloud generated from a 3D LiDAR scan. . . .	5
1.2	Long tail examples of road participant detection. . . . .	5
1.3	Road participants of the same category but with very different appearances in East (left) and West (right). . . . .	7
2.1	Knowledge system in Lifelong Learning in (Ruvolo and Eaton, 2013). . . .	18
2.2	Tug-of-War phenomenon illustration (Hadsell et al., 2020) . . . . .	19
2.3	The illustration of Hebbian Learning with Homeostatic Plasticity and Complementary Learning Systems . . . . .	20
2.4	The illustration of Elastic Weight Consolidation (EWC) . . . . .	22
2.5	The illustration of Online Elastic Weight Consolidation (Online-EWC) . . . .	23
2.6	The illustration of Learning without Forgetting (LwF) . . . . .	24
2.7	The illustration of Learning without Memorizing (LwM) . . . . .	25
2.8	The illustration of Dynamically Expandable Network (DEN) . . . . .	27
2.9	The illustration of Expert Gate . . . . .	28
2.10	The illustration of (a) Shared Bottom, (b) Mixture of Experts (MOE) and (c) Multi-Gate Mixture of Experts (MMOE) . . . . .	28
2.11	Schematic diagram of a typical multi-sensor based autonomous system for detection and tracking . . . . .	33
2.12	The illustration of Frustum PointNets . . . . .	35
3.1	Schematic diagram of catastrophic forgetting . . . . .	43
3.2	Componentized overall structure of the proposed OCL framework. . . . .	44
3.3	Example images from the left color camera in the KITTI dataset. . . . .	46
3.4	Example images from the color camera in the Waymo dataset. . . . .	47
3.5	Examples of a standard binary classification confusion matrix . . . . .	48

3.6	Intersection over Union (IoU) described in (Padilla et al., 2020).	50
4.1	Schematic diagram of 2D detection in images and corresponding 3D detection in point clouds.	54
4.2	Concept diagram of transfer learning from images to point clouds.	55
4.3	Block diagram of the OTL framework.	58
4.4	Schematic diagram of working principle of the Sample Annotator.	59
4.5	Detailed system block diagram of efficient OTL for object detection based on multimodal perception.	60
4.6	Schematic diagram of ground filtering	63
4.7	Schematic illustration of 3D object size estimation in point clouds.	73
4.8	Ablation experiment results.	74
4.9	A synchronized pair of images and point clouds from the KITTI dataset.	75
4.10	Two pairs of synchronized frames include the color image and its corresponding 3D lidar scan on KITTI.	76
4.11	A example of the detection results on the KITTI data.	76
5.1	Binary tree (top), octree (middle), and FIFO query mechanism (bottom)	84
5.2	Performance comparison of the ORF-based (online-trained) and the RF-based (offline-trained) 3D LiDAR-based object classifiers.	86
5.3	Confusion matrix for performance comparison of Integrated-ORF and Standalone-ORF.	87
5.4	Comparison of model access performance based on binary trees and octrees.	88
5.5	Changes in classification performance after deploying the Integrated-ORF learned on the KITTI dataset to the Waymo dataset and continuing to learn.	88
5.6	Confusion matrix representing the Integrated-ORF performance after learning on the Waymo dataset.	89
6.1	Schematic diagram of the Long-Short-Term Online Learning (LSTOL) framework.	96
6.2	Implementation overview of the LSTOL framework.	99

6.3 Illustrating the generation process of learning samples base on our previous work in online transfer learning (Yang et al., 2023). Rectangles denote one type of detector using point cloud which need be trained, while circles denote a well-trained detector using image. . . . . 100

6.4 Schematic diagram of Dynamic Gate Controller . . . . . 102

6.5 Confusion matrices for performance comparison of EOTL and LSTOL. For each matrix, the ordinate represents the true label, while the abscissa represents the predicted result. The darker the color, the higher the proportion of correct classifications. . . . . 108

6.6 Recall curves for cars, pedestrians, and cyclists during the system transfer from KITTI to Waymo. . . . . 109

6.7 Heatmap of “expert” opinions of each learner. Each column represents the distribution of high-confidence predictions for one learner across the test samples, while each row represents high-confidence results produced by different learners for the same class. . . . . 110



# LIST OF TABLES

2.1	Overview of All Relevant Hyperparameters . . . . .	17
2.2	Regularization-based Methods . . . . .	22
2.3	Advantages and Limitations of Multi-Sensor based Object Detection . . . .	32
2.4	Comparison in Advantages and Limitations of Cameras and 3D LiDAR: . .	34
4.1	Performance Evaluation of Deep Learning based 2D detectors . . . . .	61
4.2	Features for Classifier Training . . . . .	65
4.3	Evaluation Results on The KITTI 3D Object Detection Validation Set . . . .	70
4.4	Evaluation Results on The KITTI 2D Object Detection Validation Set . . . .	70
4.5	Evaluation Results Using Average Precision on The KITTI 3D Object De- tection Test Set (for Ranking) . . . . .	71
4.6	Evaluation Results Using Average Precision on The KITTI 2D Object De- tection Test Set (for Ranking) . . . . .	72
6.1	Evaluation of Accuracy on KITTI Dataset before and after trained on Waymo Dataset . . . . .	106





# A

## PUBLICATIONS

### A.1/ CONFERENCES

Yang, R., Yan, Z., Yang, T., Ruichek, Y. **Improved! Preventing Catastrophic Forgetting in Online Learning for Autonomous Driving**. In International Conference on Intelligent Robots and Systems (IROS), 2024, under review.

Yao, D., Liu, B., Yang, R., Yan, Z., Fu, W., Yang, T. **Few-shot Online Learning for 3D Object Detection in Autonomous Driving**. In International Conference on Autonomous Unmanned Systems (ICAUS), 2023, Nanjing, China: 1-10.

Yang, R., Yan, Z., Yang, T., Ruichek, Y. **Efficient Online Transfer Learning for 3D Object Classification in Autonomous Driving**. In IEEE International Intelligent Transportation Systems Conference (ITSC), 2021, Indianapolis, USA: 2950-2957.

### A.2/ JOURNALS

Yang, R., Yan, Z., Yang, T., Wang, Y., Ruichek, Y. **Efficient Online Transfer Learning for Road Participants Detection in Autonomous Driving**. IEEE Sensors Journal, 2023: 23(19), 23522-23535.





**Title:** Online Continual Learning for 3D Detection of Road Participants in Autonomous Driving

**Keywords:** Online Learning, Continual Learning, Object Detection, Point cloud, Multimodal perception, Autonomous driving

**Abstract:**

Autonomous driving technology has undergone rapid development in the past decade, and now both academia and industry generally concur that autonomous driving issues have converged towards machine perception. However, the environment in which autonomous vehicles are deployed is naturally complex, and even Operational Design Domain (ODD) cannot exhaust all scenarios. Consequently, models trained offline are inherently unable to support the long-term, unattended operation of autonomous vehicles. Therefore, this dissertation

studies methods for online learning models, first to answer the question of how to effectively generate online learning samples, then to explore how to effectively train models and better access them, and finally to gain insights into avoid forgetting during long-term learning. This research uses road participant detection in point clouds generated by 3D LiDAR as a downstream task to demonstrate the effectiveness of the proposed online continual learning method.

**Titre :** Apprentissage en ligne continu pour la détection en 3D des participants à la circulation routière en conduite autonome

**Mots-clés :** Apprentissage en ligne, Apprentissage continu, Détection d'objets, Nuage de points, Perception multimodale, Conduite autonome

**Résumé :**

La technologie de conduite autonome a connu un développement rapide au cours de la dernière décennie, et maintenant tant le milieu universitaire que l'industrie s'accordent généralement à dire que les problèmes de conduite autonome ont convergé vers la perception par machine. Cependant, l'environnement dans lequel les véhicules autonomes sont déployés est naturellement complexe, et même le domaine de conception opérationnelle (ODD) ne peut pas couvrir tous les scénarios. En conséquence, les modèles formés hors ligne ne sont intrinsèquement pas en mesure de prendre en charge le fonctionnement à long terme et sans surveillance des véhicules autonomes.

Par conséquent, cette thèse étudie des méthodes pour les modèles d'apprentissage en ligne, d'abord pour répondre à la question de comment générer efficacement des échantillons d'apprentissage en ligne, puis pour explorer comment former efficacement des modèles et y accéder, et enfin pour obtenir des informations sur l'évitement de l'oubli lors de l'apprentissage à long terme. Cette recherche utilise la détection des participants à la circulation routière dans des nuages de points générés par un LiDAR 3D comme tâche secondaire pour démontrer l'efficacité de la méthode proposée d'apprentissage en ligne continu.