Cooperative Sensing and Computation for Environment Perception in Autonomous Driving with Vehicular Edge Computing

by

Xuehan Ye

A thesis

presented to the University of Waterloo

in fulfillment of the

thesis requirement for the degree of

Master of Applied Science

in

Electrical and Computing Engineering

Waterloo, Ontario, Canada, 2022

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

The advances in sensing technologies, artificial intelligence, Internet of vehicles, and edge computing paradigm pave the way for autonomous driving, which is a key use case that will reshape the future transportation systems in the 5G and 6G eras. Environment perception is a key module in autonomous driving that enables the autonomous vehicles (AVs) to view the surrounding environment, facilitating situational-aware decision and planning for autonomous driving. In this work, we consider a perception task for object detection and classification, and investigate a raw data level cooperative perception scheme, with cooperative sensing among AVs and cooperative computation among both edge server and AVs, to satisfy the stringent accuracy and delay requirements for the perception task with communication and computing resource efficiency. To exploit the differentiated sensing data quality at each AV for different objects, we partition the perception task into parallel object classification subtasks, and propose a differentiated data selection strategy which selects sensing data from different AVs for each subtask with accuracy satisfaction and resource efficiency. The computation of different subtasks is distributed in a vehicular edge computing network, in a communication efficient manner. An optimization problem is formulated for joint data selection, subtask placement and resource allocation, to minimize the total communication and computing resource consumption cost, while satisfying the delay and accuracy requirements. To facilitate data selection decision with accuracy satisfaction, a deep neural network (DNN) model is pre-trained to profile an accuracy estimation function, which estimates the accuracy for each object classification subtask given the data selection decision and the sensing data quality characterized by the sensing data volume and spatial diversity. An iterative solution based on genetic algorithm is proposed for the optimization problem. Simulation results demonstrate the accuracy improvement by the cooperative sensing strategy and the

resource efficiency of the proposed differentiated data selection and subtask placement scheme.

Acknowledgements

I would like to thank all the people who made this thesis possible. First and foremost, I would like to express all my appreciation and gratitude to my supervisor Professor Weihua Zhuang for her continuous guidance, encouragement, support and care. I will always be indebted to my supervisor for providing me the opportunity to pursue my Masters degree. She was very generous on providing her invaluable advice both in research and my career. In addition to knowledge sharing, she was professional, helpful, nice, caring and always available. Professor Zhuang is definitely a shining model for the professional academic supervisor.

I would like to extend my sincere gratitude to Professor Xuemin (Sherman) Shen for organizing weekly group meetings and sharing his valuable points and experiences on how to become a qualified scientific researcher.

I would like to express my special my thanks to Kaige Qu for her great support. She is always helpful and available to discuss research issues. I would also like to thank my other colleagues at the Broadband Communication Research group: Mushu Li, Conghao Zhou, Qihao Li, Jie Gao, Wen Wu. The weekly meetings were always fruitful, which were a learning opportunity.

I am deeply indebted to my parents for their persistent motivation, support and generosity. Without their support, this thesis would not be completed.

Finally, I would like to thank all my friends both inside and outside of University of Waterloo for their company, help and support which made my life at Waterloo so pleasant.

Dedication

The thesis is dedicated to my parents, Chengkai Ye and Guoping Lin.

Table of Contents

List of Figures

List of Tables

List of Acronyms

AV	Autonomous vehicle
CNN	Convolution neural network
DC	Differentiated cooperation
DNN	Deep neural network
GA	Genetic algorithm
GPS	Global positioning system
IoU	Intersection of union
IoV	Internet of vehicle
LiDAR	Light detection and ranging
MEC	Mobile edge computing
QoS	Quality-of-service
RoI	Region of interest
SLAM	Simultaneous localization and mapping
UC	Unified cooperation
V2I	Vehicle-to-infrastructure
V2V	Vehicle-to-vehicle
V2X	Vehicle-to-everything
VEC	Vehicular edge computing

List of Symbols

A	Accuracy requirement
a_m	CNN classification accuracy for object m
$\hat{a}_m$	Predicted CNN classification accuracy for subtask m
B	Total bandwidth
C_n	Computing time at node n
$C_{n,n'}$	Transmission time over link (n, n') using whole bandwidth B
$\boldsymbol{c}_m$	3D cuboid region parameters for object m
$\hat{\boldsymbol{c}}_m$	Predicted cuboid region parameters for object m
$\mathcal{D}_n$	Raw sensing data (3D point cloud) of AV n
$\mathcal{D}_n^{(m)}$	Partial sensing data of AV n for object m
$\mathcal{D}^{(m)}$	Fused sensing data for subtask m
$d_{n,n'}$	Distance between AV n and computing node n'
$\boldsymbol{e}$	Binary subtask placement decision matrix
$e_n^{(m)}$	Binary decision variable indicating whether subtask m is placed at computing node n or not
f_n	Available computing resources (in cycle/s) at computing node n
$h_{n,n'}$	Channel fading coefficient from AV n to computing node n'
$\mathcal{J}$	Index set for J individuals in each generation

J	Population size
K	Cuboid region partition resolution in data quality vector definition
L	Number of object classes
$\mathcal{L}^{\mathtt{A}}$	Set of directed links composed of all the activated links
$l_m^{(x)}, l_m^{(y)}, l_m^{(z)}$	Lengths along x, y, and z axes of 3D cuboid region $\boldsymbol{c}_m$
$\mathcal{M}$	Object index set
M	Number of objects
m	Object index
$\mathcal{N}$	Set of AVs
$\mathcal{N}^{\mathtt{A}}$	Set of nodes composed of the starting and ending nodes (either AV or RSU) of all the activated links
$\mathcal{N}^{+}$	Set of computing nodes including all the AVs in set $\mathcal{N}$ and the RSU
$\mathcal{N}^{(m)}$	Set of AVs whose sensing data is selected for subtask m
N	Number of autonomous vehicles (AVs)
n	AV index
(n, n')	Directed link from node n to node n'
o	Total resource consumption cost for cooperative perception
$o^{\tau,j}$	Cost of individual $\mathbf{V}^{\tau,j}$
$o^*(\boldsymbol{s}, \boldsymbol{e})$	Minimal total resource consumption cost obtained by solving the resource allocation subproblem given $(\boldsymbol{s}, \boldsymbol{e})$
P_n	Uplink transmit power of AV n
$p^{\tau,j}$	Selection probability for individual $\mathbf{V}^{\tau,j}$ in generation τ

$R_{n,n'}$	Average transmission rate over the communication link from AV n to computing node n'
$\boldsymbol{s}$	Binary data selection decision matrix
$s_n^{(m)}$	Binary decision variable indicating whether the partial sensing data of AV n is selected for object m or not
T	Delay requirement
$t_{n,n'}$	Transmission time for all the sensing data transmitted from AV n to computing node n'
$\mathbf{V}^{\tau,j}$	The j-th individual in generation τ
$\boldsymbol{v}_m$	The m-th gene of an individual
$\boldsymbol{v}_m^{\tau,j}$	The m-th gene of the j-th individual in generation τ
x_m, y_m, z_m	3D location coordinates of the center of 3D cuboid region $\boldsymbol{c}_m$
$\boldsymbol{Z}^{(m)}$	Fused data quality vector for subtask m
$\boldsymbol{Z}_n^{(m)}$	Data quality vector for partial sensing data $\mathcal{D}_n^{(m)}$
$Z_{n,k}^{(m)}$	Number of observation points located inside the k-th 3D cuboid sub-region in partial sensing data $\mathcal{D}_n^{(m)}$
$\boldsymbol{\alpha}$	Continuous computing resource usage decision vector
$\boldsymbol{\alpha}^{\mathtt{A}}$	Continuous computing resource usage decision variables for computing nodes in set $\mathcal{N}^{\mathtt{A}}$
α_n	Fraction of computing resource usage at computing node n
$\boldsymbol{\beta}$	Continuous bandwidth allocation decision matrix
$\boldsymbol{\beta}^{\mathtt{A}}$	Continuous bandwidth allocation decision variables for links in set $\mathcal{L}^{\mathtt{A}}$

$\beta_{n,n'}$	Fraction of bandwidth allocated to the communication link from AV n to computing node n'
γ	Path loss exponent
$\delta_m^{(x)}, \delta_m^{(y)}, \delta_m^{(z)}$	Rotation angles along x, y, and z axes of 3D cuboid region $\boldsymbol{c}_m$
ϵ	Computation intensity (in cycle/point)
$\kappa\left(\boldsymbol{s}, \boldsymbol{e}\right)$	Feasibility of the resource allocation subproblem given $(\boldsymbol{s}, \boldsymbol{e})$
μ_n	Computing demand in CPU cycles for all the subtasks placed at computing node n
$\mu^{(m)}$	Computing demand in CPU cycles for subtask m
$\rho_{n,n'}$	Total size of the sensing data transmitted over the communication link from AV n to computing node n'
σ^2	Noise power
$\varsigma^{\tau,j}$	Feasibility of individual $\mathbf{V}^{\tau,j}$
τ	Generation index
$\boldsymbol{\Phi}^{\tau}$	Population in generation τ
φ	Data size (in bit) of one observation point
$\boldsymbol{\chi}$	Auxiliary binary link activation decision matrix
$\chi_{n,n'}$	Binary decision variable indicating whether the wireless communication (either V2I or V2V) link between AV n and computing node n' is activated or not
$\boldsymbol{\psi}^{\mathtt{A}}$, $\boldsymbol{\theta}^{\mathtt{A}}$, ξ	Auxiliary continuous decision variables
ω	Weighting factor in total resource consumption cost o

Chapter 1

Introduction

1.1 Perception task

Autonomous driving is a key use case that will reshape the future transportation systems in the 5G and 6G era. The foundation of autonomous driving is the capability for vehicles to know their surrounding environments and maintain real-time situational awareness under environmental dynamics, referred to as environment perception, based on which different autonomous driving applications can be supported, such as localization, motion control, and path planning [4,7,31,53,56]. An autonomous vehicle (AV) is equipped with various onboard sensors such as cameras, light detection and ranging (LiDAR) sensors, and radio detection and ranging (radar) sensors, for different perception tasks, e.g., object detection, classification, orientation estimation and tracking. Here, we consider a perception task for object detection and classification, which is to detect and classify objects in the region of interest (RoI) of an AV at a given time instant, based on which the AV can infer orientations and trajectories of surrounding objects. The

object orientations and trajectories are continually updated over time, with a time-varying updating frequency depending on the overall environmental dynamics [19, 37]. For each update, a new perception task for object detection and classification is initiated by the AV. The perception task is executed in two phases, referred to as object detection phase and object classification phase respectively [7, 60]. In object detection phase, the existence of unknown objects in a considered RoI is detected and the bounding boxes for all the detected objects are estimated. In object classification phase, the class of each detected object is identified, usually based on advanced deep learning techniques. For real-time situational awareness, the perception task has a stringent delay requirement in milliseconds (e.g., 10-20 ms), and the accuracy requirement for object detection and classification should be satisfied.

As the sensors for environment perception typically rely on line-of-sight (LOS) sensing, their sensing range is fragile and easy to be blocked by surrounding obstacles [53]. For example, when an inter-vehicle distance becomes too short, the LOS area of an AV shrinks due to the blockage by a neighbor vehicle, making some nearby objects whose status is critical for autonomous diving totally invisible [27]. Moreover, it is beneficial to have sensing data from diverse point of views of an object for accurate classification. For example, a cyclist viewed solely from the front has a high chance to be falsely classified as a pedestrian due to feature similarity [53]. If sensing data for more sides of a cyclist is used for classification, the probability of false classification as a pedestrian becomes much lower. However, the onboard sensors of a single vehicle only provide limited sensing data diversity for an object even at a time instant without sensor occlusion, as the vehicle observes the object from a limited viewpoint. Hence, we cannot fully rely on the onboard sensors for a consistent guarantee of complete and accurate environment perception especially in a highly obstructed environment.

1.2 Cooperative perception

With the emerging Internet of vehicles (IoV) technologies, vehicles are connected with each other and with infrastructures via vehicle-to-everything (V2X) communication, which provides an opportunity for augmented perception capability through cooperative sensing [4, 22, 27]. Cooperative sensing exploits V2X communication to enable sharing of the raw or processed sensing data among vehicles and infrastructure in proximity, by which the perception range can be extended beyond line of sight and field of view, and the perception accuracy can be enhanced [23, 43]. This augmented perception capability contributes to better decision making and planning for autonomous driving, such as safe lane changing, smooth braking/acceleration, and hidden obstacle avoidance [23, 27].

Traditional cooperative sensing has been designed to assist human drivers for enhanced safety by exchanging alarm messages. It allows vehicles to exchange their processed sensing data with small data size via periodic broadcasting. For example, a vehicle processes its own raw sensing data, and generates a continually updated track list for every detected objects in its surrounding environment, which is periodically broadcast to vehicles in its proximity via vehicle-to-vehicle (V2V) communication [21]. Then, each vehicle can fuse its own object list with all the objects lists received from nearby vehicles, referred to as decision-level fusion, which may extend the object list and improve the detection accuracy for some objects. However, the perception performance enhancement with decision-level cooperative sensing is limited by the per-vehicle sensing capability. For example, the undetected objects by all vehicles remain undetected after the decision-level fusion, which cannot satisfy the high reliability requirement of autonomous driving applications [11, 43].

Recent studies have demonstrated the perception performance enhancement by cooperative sensing based on raw sensing data fusion which retains the original environmental information collected by each vehicle [11,22,27]. The raw sensing data has a much larger data size compared with decision-level object lists. For example, a typical commercial LiDAR sensor using 64 laser diodes produces 2.8 million data points per second [2]. Due to the large data size of raw sensing data, the traditional broadcast-based cooperative sensing becomes extremely communication inefficient [2, 8]. Moreover, as the fused raw sensing data should be processed using advanced artificial intelligence (AI) models such as VoxelNet [63] and PointNet [38] for accurate object classification, the local computing capability at a single vehicle cannot always support real-time AI processing for the classification of multiple objects in the environment, especially when the number of objects is large.

1.3 Motivation and Objective

With mobile edge computing (MEC), the computation-intensive perception task can be offloaded to a resource-rich edge server, where the raw sensing data transmitted from multiple AVs via vehicle-to-infrastructure (V2I) communication is fused and processed by powerful AI models with improved delay performance [10, 27, 57]. Specifically, object-wise sensing data segments are first extracted from the fused sensing data based on object detection. Then, for each object, the corresponding sensing data segment in the fused sensing data are processed by an AI model for object classification. In this edge-assisted cooperative sensing approach, each participating AV sends its whole raw sensing data to the edge server, and the sensing data segment for each object is a fusion of the corresponding data from the same group of AVs, referred to as object-

level unified cooperation. However, due to different spatial relationships between an AV and different objects, an AV provides differentiated sensing data quality for different objects. For example, an AV provides sparse sensing data for an object at a distance, and provides limited sensing data diversity for an object which is partially obstructed from the view of the AV. Hence, for each object, there exists a minimal set of cooperating AVs that should provide the sensing data for accurate classification of the object, which is different among objects. If unified cooperation is employed for the classification of different objects in a perception task, the unified cooperating AV set should be the union of the minimal cooperating AV sets of each object for accuracy satisfaction. For the objects whose minimal cooperating AV set is smaller than the unified cooperating AV set, the accuracy is over-provisioned due to providing more sensing data than required at the cost of more communication and computing resource consumption.

Therefore, we should investigate a resource efficient cooperative sensing strategy with differentiated cooperation among AVs for different objects, referred to as object-level differentiated cooperation, by selecting a different group of AVs to provide the minimum amount of sensing data for each object, which reduces both the communication and computing resource demand while satisfying the accuracy requirement. Moreover, the communication and computing efficiency for cooperative sensing can be further enhanced by distributing partial computing load among multiple AVs in proximity through V2V communication, which is enabled by the vehicular edge computing (VEC) paradigm [30, 32, 40]. The VEC integrates MEC into IoV, and creates a distributed computing hierarchy with computing resources distributed among both edge servers and vehicles. Specifically, as the AI processing for object classification can be parallelized among multiple objects, we can partition a perception task into object-wise subtasks and place the subtasks in the distributed VEC platform. Each subtask can be placed at a different

computing node which is either an edge server or an AV with available computing resources. For each subtask, all the selected sensing data from a unique group of AVs should be transmitted to the placed computing node via either V2I or V2V communication for fusion and AI processing. In this way, an AV which initiates a perception task can be assisted by nearby AVs in terms of both sensing and computing, as the nearby AVs can participate in not only cooperative sensing by sharing the sensing data but also cooperative computation by sharing the computing resources. The benefit of distributed computing for cooperative sensing is three folded. First, by moving some computation to adjacent vehicles, VEC alleviates computing load at the edge server which is not as powerful as a cloud server. Second, some sensing data are sent to vehicles in proximity via V2V communication, which is more communication efficient. Third, if an AV is selected to provide sensing data for a subtask placed at itself, no communication is required for the sensing data, thus further reducing the total communication demand.

In this thesis, a cooperative perception problem in autonomous driving is studied, to meet the real-time processing requirement and achieve perception accuracy satisfaction with enhanced communication and computing resource efficiency. The contributions are summarized as follows.

- A communication and computing resource efficient cooperative sensing and computing strategy is proposed for perception tasks in autonomous driving. Specifically, we consider a perception task for object detection and classification initiated by an AV. For cooperative sensing, an object-level differentiated cooperation scheme is proposed, which enables differentiated data selection from a different group of AVs for the classification of each object, and reduces the total communication and computing demand by reducing the sensing data amount required for accuracy satisfaction. For cooperative computation, the computing for

the classification of multiple objects is parallelized and distributed in a VEC-enabled hierarchical computing platform, which further improves the communication and computing resource efficiency;

- To facilitate the differentiated cooperative sensing strategy with high accuracy requirement, an accuracy model which characterizes the relationship between data selection decision and object classification accuracy is required. However, no relevant studies have been done to establish such an accuracy model. Hence, we propose a deep learning approach to develop an accuracy model, by which the classification accuracy for an object can be estimated based on data selection decision and the sensing data quality at each AV. To characterize the data quality of sensing data, a data quality vector is defined to capture sensing data amount and diversity, both of which affect the classification accuracy;

- A joint data selection, subtask placement and resource allocation problem is formulated as an optimization problem, which determines 1) the differentiated data selection among AVs for each object classification subtask, 2) the subtask placement among the computing nodes including both edge server and AVs, and 3) the transmission resource allocation among different V2I/V2V links and the computing resource usage at each computing node, to minimize the total transmission and computing resource consumption while satisfying the delay and accuracy requirements. A genetic algorithm based solution is proposed, which iteratively updates the data selection and subtask placement decision until convergence, based on the feasibility of an resource allocation subproblem and the minimal resource consumption cost obtained by solving the resource allocation subproblem.

1.4 Outline of the Thesis

The rest of this thesis is organized as follows. The literature review is presented in Chapter 2 and the system model is presented in Chapter 3. Chapter 4 presents a proposed DNN accuracy model, a problem formulation for joint data selection, subtask placement and resource allocation and the corresponding solution. Simulation results are presented in Chapter 5, and conclusions are drawn in Chapter 6.

Chapter 2

Literature Survey

Th chapter reviews existing works in cooperative sensing.

2.1 Decision-level Cooperative Sensing

Traditional cooperative sensing allows participating AVs to share alarm messages via periodic broadcasting, which is called decision-level cooperative sensing [21]. The alarm messages are processed by AVs individually based on their own sensing data, mainly describing the state information (e.g., location, size and direction estimates with corresponding covariances) of all objects in the AVs' surrounding environments. Related standardized messages include safety messages (BSMs) [29,34], cooperative awareness messages (CAMs), and cooperative perception messages (CPMs) [44, 50]. Then, each AV fuses its own object list with all the objects lists received from nearby AVs. The strategy may extend the object list and improve the detection accuracy for some objects. For example, if a front AV fully occludes some objects of interest of

its behind AV, it can transmit the state information of these objects to the AV behind to extend the its object list.

Communication efficiency is another issue for decision-level cooperative sensing. Many redundant alarm messages can increase the risk that important messages are delayed or even lost, potentially leading to serious safety threats. A typical way to adjust the network load is to filter the number of objects in alarm messages. For instance, each AV can intelligently select an object sublist to be sent at each transmission opportunity depending on each object's anticipated value [18]. The value of an object is defined as the relative information entropy of its posterior knowledge with respect to its prior knowledge. One object is included in the next alarm message only when its value is larger than a pre-defined threshold. Moreover, in order to avoid the network congestion, the frequency of alarm message transmissions should be regulated based on the channel load [15] [6]. To guarantee fair channel resource sharing among AV, the work in [55] further investigates a control agent for each AV that coordinates with its neighboring agents to determine the transmission rate.

2.2 Raw Data-level Cooperative Sensing

Complete safety is difficult to be achieved by decision-level cooperative sensing, as alarm messages discard many environmental details due to the restricted message size. For example, if two AVs both fail to detect one object, the fusion of their perception results still fails to detect such an object [11, 43]. Therefore, raw data-level cooperative sensing is proposed in which each participating AV shares raw sensing data [11, 22, 27]. Then, the object lists are generated by each AV based on the fused sensing data from all vehicles. Different from decision-level coop-

erative perception, raw data-level cooperative perception can fully exploit sensing data without information loss.

As the size of raw sensing data is large (e.g., a typical commercial LiDAR sensor produces 2.8 million data points per second [2]), the traditional broadcast-based raw data-level cooperative sensing is extremely communication inefficient [2, 8]. To optimize the tradeoff between communication resource consumption and perception performance, one approach is to intelligently select parts of sensing data for transmission. For instance, as objects close to AVs are likely to be detected accurately based on their own sensing data, each AV can share only the sensing data far from the AVs (i.e., outside a circle of the given radius in the horizontal plane) to further improve perception accuracy. In addition, AVs should be intelligently selected for raw sensing data sharing. When the network is congested, only the most beneficial AVs for perception performance enhancement are selected. A related strategy is designed in [27]. As a front AV usually decreases the LOS area of its behind AV, it is proposed to activate the raw sensing data transmission from the front AV to the behind AV when their inter-vehicle distance is smaller than a threshold.

Futhermore, raw data-level cooperative sensing incurs huge computation overhead as each AV has to process a large amount of raw sensing data from all AVs using advanced artificial intelligence (AI) models such as VoxelNet [63] and PointNet [38]. One typical strategy to alleviate the computing load on AVs is to deploy a computation-intensive perception task at a resource-rich edge server and require all participating AVs to transmit their raw sensing data to the edge server [10, 27, 57]. The processed perception results based on the fused sensing data are then disseminated to all AVs.

2.3 Summary

Th chapter separatively reviews existing works in decision-level and raw data-level cooperative sensing. In the decision-level cooperative perception, the traditional broadcast-based strategy is introduced and related strategies to adjust the network load are reviewed. Then in the raw data-level cooperative perception, we introduce the traditional broadcast-based strategy, related strategies to optimize the tradeoff between communication and computing resource consumption and perception performance.

Chapter 3

System Model

3.1 Autonomous Driving Scenario

As shown in Fig. 3.1, we consider an edge-assisted autonomous driving scenario over a unidirectional urban road segment under the coverage of one road side unit (RSU). The RSU is co-located with an edge server. Consider N autonomous vehicles (AVs) traveling along the road segment, including one target AV which initiates a perception task for situational awareness and $N-1$ assisting AVs which can cooperate with the target AV for perception. Let $\mathcal{N} = \{0, \cdots, N-1\}$ denote the set of AVs, with $n \in \mathcal{N}$ representing the AV index. Specifically, AV 0 corresponds to the target AV, which requires the environmental information in a region of interest (RoI). In general, the region of interest (RoI) of target AV can have a customized shape according to the application scenario, e.g., a circular or rectangular area around or in the driving direction of the target vehicle, for modeling purposes [2, 53]. Here, we consider an RoI spreading over the the front area of the target vehicle as illustrated in Fig. 3.1 [2], as more more focus is placed on the

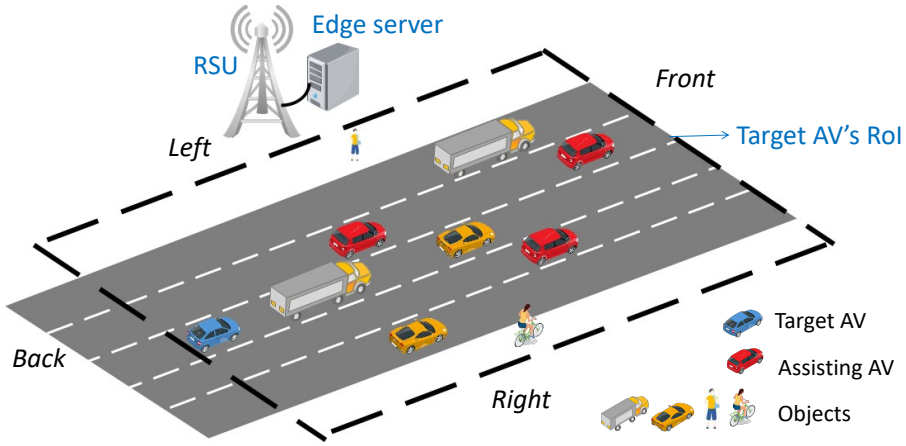


Figure 3.1: An illustration of edge-assisted autonomous driving scenario.

sensing capability in the driving direction of the target vehicle. The range of RoI depends on the vehicle speed and time of interest [53]. For example, if the target AV with a speed of $60\ km/h$ is interested in the environment which is 3 seconds ahead of its driving direction, the length of its RoI is $50\ m$. The proposed cooperative sensing and computation mechanism can be applied to an autonomous driving scenario with an RoI in a general shape.

3.2 Object Model

Let integers M and m denote the total number of unknown objects in the target AV's RoI and the object index respectively, with $0 \leq m \leq M-1$. Let set $\mathcal{M}$ contain all object indexes, i.e., $\mathcal{M} = \{0, \cdots, M-1\}$. Let L denote the number of object classes. In Fig. 3.1, we have $L = 4$ classes of objects including car, truck, pedestrian and cyclist. Note that both the car and truck objects are non-autonomous vehicles with human drivers. The perception task of the target AV is to 1) detect the existence and spatial location of all the M objects in the RoI,

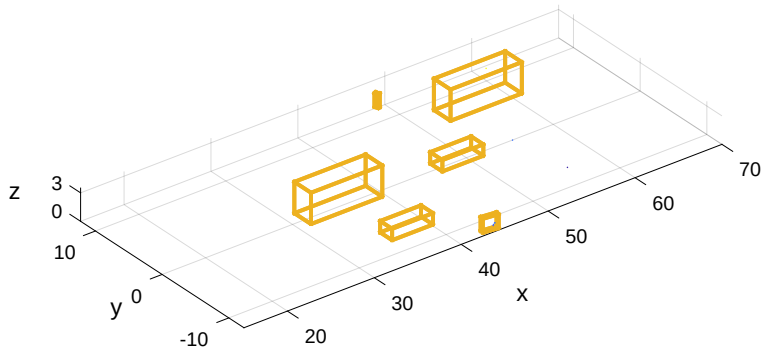


Figure 3.2: An illustration of cuboid regions for six objects in target AV's RoI.

and 2) estimate the class for each detected object m. The spatial location of object m can be represented by a 3D cuboid region containing the object, which is characterized by a 9-tuple, $\boldsymbol{c}_m = \left(x_m, y_m, z_m, l_m^{(x)}, l_m^{(y)}, l_m^{(z)}, \delta_m^{(x)}, \delta_m^{(y)}, \delta_m^{(z)}\right)$. Here, x_m, y_m, z_m specify the 3D location coordinates of the cuboid center, $l_m^{(x)}, l_m^{(y)}, l_m^{(z)}$ specify the lengths of the cuboid along the x, y, and z axes, and $\delta_m^{(x)}, \delta_m^{(y)}, \delta_m^{(z)}$ specify the rotation angles for the cuboid along the x, y, and z axes. Note that the coordinate parameters, x_m, y_m, z_m, are all defined in the global coordinate system, which can be obtained by coordinate transformation between local and global coordinate systems [22].

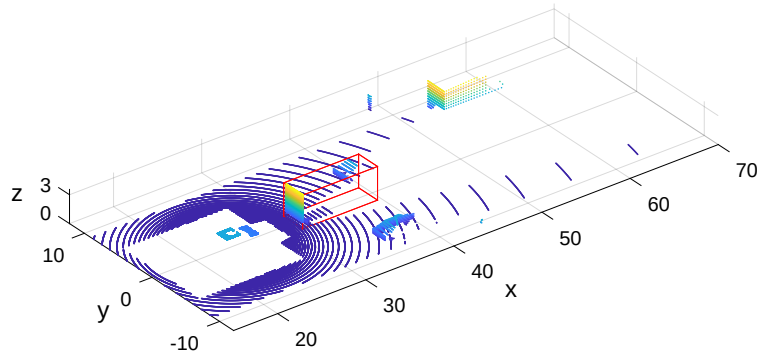
Without loss of generality, we assume zero rotation angles for all the objects. Then, 3D cuboid region $\boldsymbol{c}_m$ for object m can be characterized by a 6-tuple, $\boldsymbol{c}_m = \left(x_m, y_m, z_m, l_m^{(x)}, l_m^{(y)}, l_m^{(z)}\right)$, for simplicity. Fig. 3.2 depicts the 3D cuboid regions with zero rotation angles for $M = 6$ objects in the target AV's RoI. Each 3D cuboid region is indicated by a yellow box.

3.3 Sensing Data Model

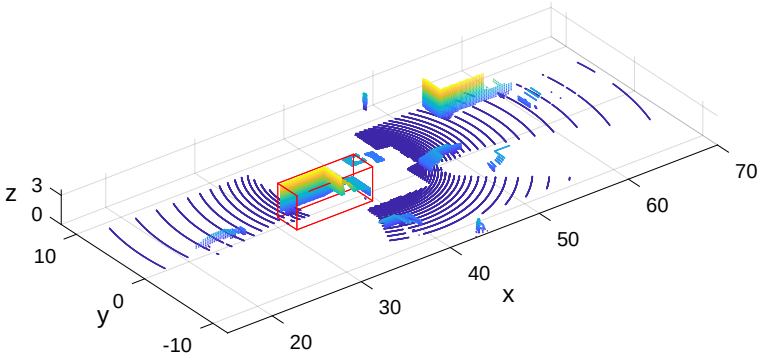
2D monocular images are usually used by AVs for object classification. However, due to indeterministic locations of 3D image points projected by 2D pixels, it is difficult to fuse multiple 2D monocular images shared by spatial diverse AVs. Therefore, we consider 3D point clouds, which are sets of deterministic 3D point locations, for cooperative perception. The data can be generated by light detection and ranging (LiDAR) sensors or depth cameras [5]. Especially, 3D point clouds transformed by depth camera images can provide rich enough details for accurate object classification. Denote the 3D point cloud produced by AV n by $\mathcal{D}_n = \{(x_n^i, y_n^i, z_n^i)\}$, which is a set of 3D location coordinates of different observation points in the environment. Each element of 3D sensing data $\mathcal{D}_n$, (x_n^i, y_n^i, z_n^i), corresponds to one observation point, e.g., an observation point on the surface of an object. The total number of observation points in sensing data $\mathcal{D}_n$ of AV n is denoted by $|\mathcal{D}_n|$. Note that the observation point location coordinates for different AVs are all aligned with the global coordinate system via coordinate transformation, as each AV's location can be obtained accurately by the global positioning system (GPS) or the simultaneous localization and mapping (SLAM) techniques [12].

Once the six parameters characterizing 3D cuboid region $\boldsymbol{c}_m$ for object m are known, we can extract a subset of observation points for object m from $\mathcal{D}_n$, denoted by $\mathcal{D}_n^{(m)}$, which is the partial sensing data of AV n for object m. An observation point with 3D location coordinates (x_n^i, y_n^i, z_n^i) is included in subset $\mathcal{D}_n^{(m)}$ if the point is located inside 3D cuboid region $\boldsymbol{c}_m$, represented as

$$\begin{cases} x_m - \frac{l_m^{(x)}}{2} \leq x_n^i \leq x_m + \frac{l_m^{(x)}}{2} \\ y_m - \frac{l_m^{(y)}}{2} \leq y_n^i \leq y_m + \frac{l_m^{(y)}}{2} \\ z_m - \frac{l_m^{(z)}}{2} \leq z_n^i \leq z_m + \frac{l_m^{(z)}}{2}. \end{cases} \tag{3.1}$$

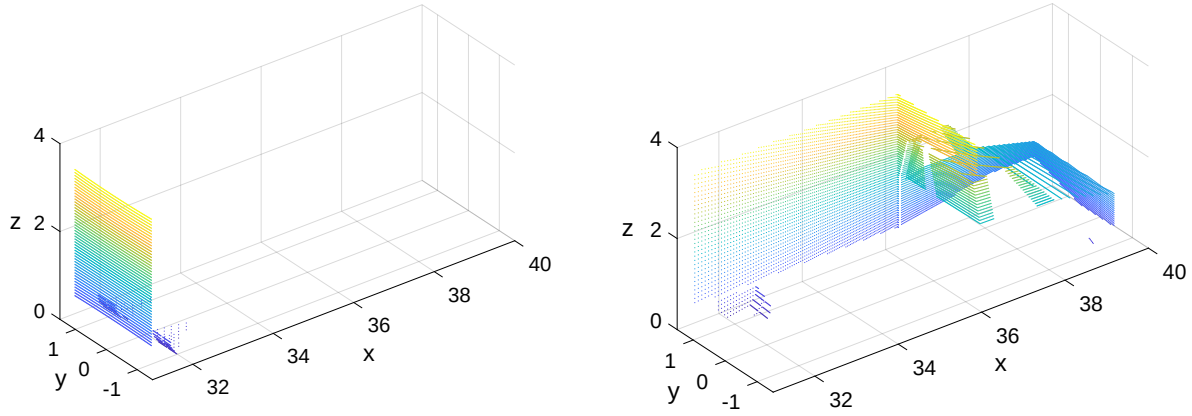


(a) Provided by the target AV



(b) Provided by an assisting AV

Figure 3.3: Point clouds of two AVs.

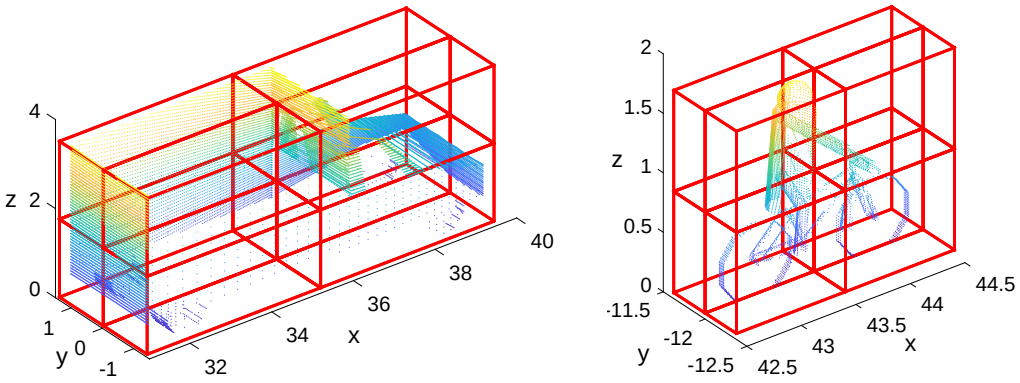


(a) Provided by the target AV

(b) Provided by an assisting AV

Figure 3.4: Partial sensing data of two AVs for a truck.

If object m is not in the sensing range of AV n due to distance or occlusion, the corresponding partial sensing data, $\mathcal{D}_n^{(m)}$, is an empty set. As different AVs provide different views from different distances and angles to an object, the partial sensing data of different AVs for the same object contains a different number of observation points and shows different spatial distribution. Fig. 3.3 illustrates the sensing data (i.e., LiDAR point clouds) generated by the target AV and an assisting AV in the network scenario shown in Fig. 3.1. Specifically, the assisting AV is at the front left corner of a truck in front of the target AV. A 3D cuboid region containing the truck is indicated by a red box in Fig. 3.3. With such a 3D cuboid region, the partial sensing data corresponding to the truck is extracted from both point clouds and shown in Fig. 3.4. It is observed that the observation points of the target AV for the truck are concentrated at the back side of the truck, while the observation points of the assisting AV for the truck mainly spread over the front and left sides. Fig. 3.4 (b) also shows that the observation points closer to the back of the truck are more sparse due to longer distance from the assisting AV.



(a) For a truck

(b) For a cyclist

Figure 3.5: An illustration of cuboid region partition with cuboid region partition resolution $K = 2$.

Data Quality Vector

To coarsely characterize the spatial distribution of partial sensing data $\mathcal{D}_n^{(m)}$ of AV n for object m, we partition 3D cuboid region $\boldsymbol{c}_m$ for object m into K^3 disjoint 3D cuboid sub-regions, where K denotes the partition resolution corresponding to the number of partitions along each axis. The lengths of each 3D cuboid sub-region along the x, y, and z axes for object m are $\frac{l_m^{(x)}}{K}$, $\frac{l_m^{(y)}}{K}$ and $\frac{l_m^{(z)}}{K}$, respectively. Fig. 3.5 illustrates the 3D cuboid region partition for both a truck and a cyclist with $K = 2$. There are many factors affecting the object classification performance, e.g., the distance and angle between the sensor and object, and hardware noises of sensors. For simplicity, we consider only two main factors to estimate object classification performance, i.e., the distance and angle between the sensor and object, under the assumption that all the sensors have the same measurement accuracy. Note that, in practice, the sensing data accuracy can change significantly with the distance and angle between the sensor and object, which should be taken into account for maximal performance in object classification. For an object, AVs with different relative distances and angles generate sensing data with different point numbers and spatial distributions.

Therefore, we characterize the data quality $\boldsymbol{Z}_n^{(m)} \in \mathbb{R}^{K^3}$ for partial sensing data $\mathcal{D}_n^{(m)}$ in terms of point number and spatial distribution. The k-th ($1 \leq k \leq K^3$) element, $Z_{n,k}^{(m)}$, in vector $\boldsymbol{Z}_n^{(m)}$ denotes the number of observation points located inside the k-th 3D cuboid sub-region among all the observation points in partial sensing data $\mathcal{D}_n^{(m)}$. If the observation points in partial sensing data $\mathcal{D}_n^{(m)}$ are more spatially distributed inside 3D cuboid region $\boldsymbol{c}_m$, more 3D cuboid sub-regions corresponding to $\boldsymbol{c}_m$ contain observation points by AV n on the surface of object m, and data quality vector $\boldsymbol{Z}_n^{(m)}$ contains more non-zero elements. Typically, if the sensing data for object m has a data quality vector with more non-zero elements and with a larger value for each non-zero element, there are more observation points spreading over the surface of the object, leading to a higher object classification accuracy.

A data quality vector is also defined for the fused partial sensing data of multiple AVs corresponding to the same object. For example, if the partial sensing data of AV n and AV n' for object m are fused, the fused sensing data are $\mathcal{D}_n^{(m)} \cup \mathcal{D}_{n'}^{(m)}$, with a data quality vector of $\boldsymbol{Z}_n^{(m)} + \boldsymbol{Z}_{n'}^{(m)}$. As AV n and AV n' observe object m from different distance and angle, the fused sensing data, $\mathcal{D}_n^{(m)} \cup \mathcal{D}_{n'}^{(m)}$, potentially contains more observation points which are more spatially distributed over the surface of object m, leading to an improved object classification accuracy. Correspondingly, vector $\boldsymbol{Z}_n^{(m)} + \boldsymbol{Z}_{n'}^{(m)}$ potentially contains more non-zero elements especially if there are more non-zero elements in vector $\boldsymbol{Z}_n^{(m)}$ and vector $\boldsymbol{Z}_{n'}^{(m)}$ at different positions. If both vectors have a non-zero element at the same position, the corresponding element in $\boldsymbol{Z}_n^{(m)} + \boldsymbol{Z}_{n'}^{(m)}$ has a larger value.

3.4 Task Model

The perception task of the target AV is executed in an object detection phase and an object classification phase sequentially.

Object Detection Phase

The goal of object detection is to successfully detect the existence of all the M unknown objects in the RoI and have an accurate cuboid region estimation for each object. The accuracy of cuboid region estimation is usually evaluated by an intersection over union (IoU) metric, which is the ratio between area of overlap and area of union for the ground-truth and estimated cuboid regions. Typically, object detection is not highly resource demanding, as periodical broadcasting by sharing low-resolution sensing data [5] or efficient quadtree data structures [2] can provide sufficient accuracy for object detection. CarSpeak is proposed to utilize sensing data quality to further improve the resource efficiency for object detection [24]. Specifically, it uses quadtree data structures to represent sensor information, defines the information quality of each region (corresponding to a sub-tree in quadtree data structures) based on completeness and freshness, and provides each AV a share of the medium proportional to the quality of information it possesses.

We adopt existing cooperative perception methods in object detection phase which have been extensively studied. For simplicity, we obtain the estimated cuboid region parameters with the following operations among target and assisting AVs, as illustrated in Fig. 3.6, again under the simplified assumption that all the sensing data have the same measurement accuracy. First, each AV generates a 3D point cloud as raw sensing data. By down-sampling the raw sensing data

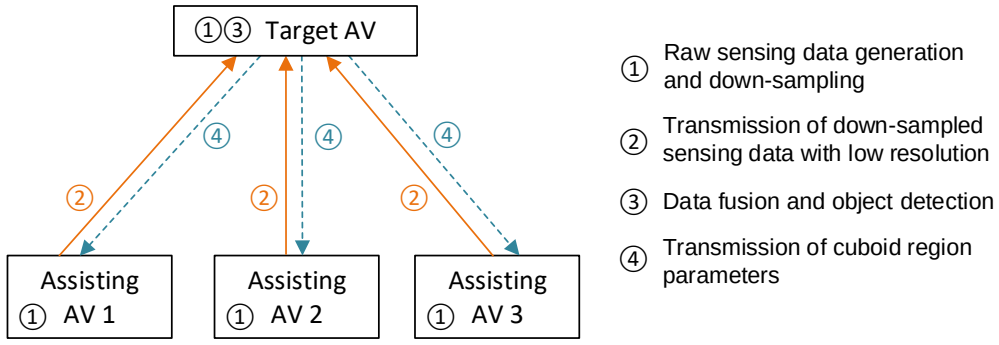


Figure 3.6: Operations among target and assisting AVs in object detection phase for $N = 4$.

with a given down-sampling ratio, low-resolution sensing data are obtained with reduced data resolution. Then, each assisting AV transmits the low-resolution sensing data to the target AV. After the target AV receives the low-resolution sensing data from all the assisting AVs, it fuses all the received data with its own low-resolution sensing data, and executes a classic object detection algorithm such as the Euclidean clustering and L-shape fitting algorithm based on the fused low-resolution sensing data [58, 64]. The output of the object detection algorithm is a group of estimated 6-tuple cuboid region parameters. Each 6-tuple corresponds to one object in the RoI. Finally, the target AV transmits the estimated cuboid region parameters to all the assisting AVs. Broadcast transmission can be used in the whole object detection phase, as the transmission data size is small.

Object Classification Phase

Object classification is usually performed by a convolution neural network (CNN), which estimates the probabilities of each class for an object based on the point location coordinates in the 3D point cloud. Consider the estimated probability for the true class of object m as the classification accuracy for object m, denoted by a_m. Let A denote the accuracy requirement for the

classification of each object in the RoI. Typically, object classification requires high-resolution sensing data with large size for good accuracy, which is computation intensive [62]. Here, we use raw sensing data for object classification. Let T denote the delay requirement for the perception task in object classification phase. Then, it is required that the classification for any object m should be completed with accuracy $a_m \geq A$ in a time duration not exceeding T.

Without loss of generality, we assume that all the M objects are successfully detected in the object detection phase, and the estimated cuboid region for object m is sufficiently accurate to contain the whole object. Let $\hat{\boldsymbol{c}}_m = \left(\hat{x}_m, \hat{y}_m, \hat{z}_m, \hat{l}_m^{(x)}, \hat{l}_m^{(y)}, \hat{l}_m^{(z)}\right)$ denote the estimated cuboid region parameters for object m, based on which the partial sensing data of AV n for object m, i.e., $\mathcal{D}_n^{(m)}$, can be extracted from the whole sensing data, $\mathcal{D}_n$, of AV n, according to (3.1). The data quality vector for partial sensing data $\mathcal{D}_n^{(m)}$, i.e., $\boldsymbol{Z}_n^{(m)} \in \mathbb{R}^{K^3}$, can be calculated for a given cuboid region partition resolution K, based on the definition in Subsection 3.3.

With the object-specific partial sensing data for different objects, we can partition the perception task in object classification phase into M subtasks, among which subtask m is to classify object m with accuracy $a_m \geq A$ within delay bound T.

For subtask m, there are at most N sets of available partial sensing data provided by N AVs. If the corresponding partial sensing data of multiple AVs are used for subtask m, the sensing data should be fused before being processed by a CNN classification model. Let $\boldsymbol{s} = \{s_n^{(m)}, \forall n \in \mathcal{N}, \forall m \in \mathcal{M}\}$ denote a binary data selection decision matrix in $\mathbb{R}^{N \times M}$, with decision variable $s_n^{(m)} = 1$ indicating that the partial sensing data of AV n for object m, i.e., $\mathcal{D}_n^{(m)}$, is used for subtask m, and $s_n^{(m)} = 0$ otherwise. Let $\mathcal{N}^{(m)} = \left\{n \in \mathcal{N} \middle| s_n^{(m)} = 1\right\}$ denote the set of AVs whose corresponding partial sensing data are selected for subtask m. Then, the fused sensing data for subtask m, denoted by $\mathcal{D}^{(m)}$, are a union of all the selected partial sensing data from

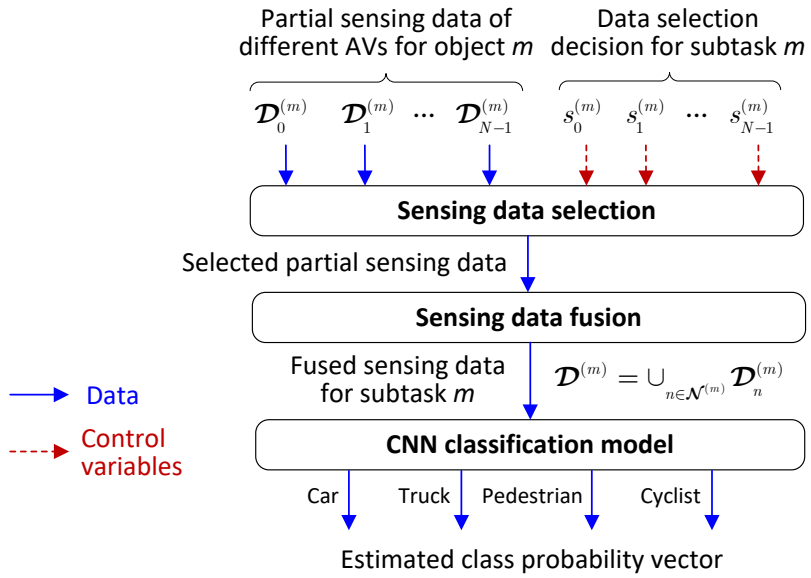


Figure 3.7: A workflow of sensing data selection, fusion and CNN classification for subtask m with $L = 4$.

AVs in $\mathcal{N}^{(m)}$, given by

$$\mathcal{D}^{(m)} = \cup_{n \in \mathcal{N}^{(m)}} \mathcal{D}_n^{(m)}. \tag{3.2}$$

Let $\boldsymbol{Z}^{(m)}$ be a data quality vector which characterizes the data quality of fused sensing data $\mathcal{D}^{(m)}$ for subtask m, referred to as fused data quality vector for subtask m. As defined in Subsection 3.3, fused data quality vector $\boldsymbol{Z}^{(m)}$ for subtask m is calculated as

$$\boldsymbol{Z}^{(m)} = \sum_{n \in \mathcal{N}} s_n^{(m)} \boldsymbol{Z}_n^{(m)} \tag{3.3}$$

where $\boldsymbol{Z}_n^{(m)}$ is the data quality vector for partial sensing data $\mathcal{D}_n^{(m)}$.

For subtask m, fused sensing data $\mathcal{D}^{(m)}$ is processed by a CNN classification model, which

generates an L-dimension estimated class probability vector for subtask m, where L is the number of object classes. Fig. 3.7 shows a workflow of sensing data selection, fusion and CNN classification for subtask m with $L = 4$.

3.5 Computing Model

With vehicular edge computing, the perception task of the target AV can be processed locally at the target AV or remotely at the RSU and assisting AVs in proximity. Note that the RSU is co-located with an edge server, thus providing computing capability. Let $\mathcal{N}^+ = \mathcal{N} \cup \{N\}$ denote the set of computing nodes including all the AVs in set $\mathcal{N}$ and the RSU which is referred to as computing node N. If AV n serves as a computing node for any subtask, we refer to AV n as computing node n. Let f_n denote the amount of available computing resources (in cycle/s) at computing node $n \in \mathcal{N}^+$. Here, we focus on the computing model in the object classification phase, with the consideration of high computing demand for object classification.

As mentioned, the perception task in object classification phase is partitioned into M subtasks, each for an object. Each subtask is placed at a computing node. Let $\boldsymbol{e} = \{e_n^{(m)}, \forall n \in \mathcal{N}^+, \forall m \in \mathcal{M}\}$ denote a binary subtask placement decision matrix in $\mathbb{R}^{(N+1)\times M}$, with decision variable $e_n^{(m)} = 1$ indicating that subtask m is placed at computing node n, and $e_n^{(m)} = 0$ otherwise. Subtask m must be placed at a single computing node, given by

$$\sum_{n \in \mathcal{N}^+} e_n^{(m)} = 1, \quad \forall m \in \mathcal{M}. \tag{3.4}$$

If $e_n^{(m)} = 1$, all the selected partial sensing data for subtask m from different AVs should be fused

and then computed at computing node n.

Let ϵ denote the computation intensity (in cycle/point) which is the average number of CPU cycles for computing one observation point in sensing data. Then, the computing demand in CPU cycles for subtask m, denoted by $\mu^{(m)}$, depends on the total number of observation points in all the selected partial sensing data from different AVs, given by

$$\mu^{(m)} = \epsilon \sum_{n \in \mathcal{N}} s_n^{(m)} \left| \mathcal{D}_n^{(m)} \right|, \quad \forall m \in \mathcal{M} \tag{3.5}$$

where $\left| \mathcal{D}_n^{(m)} \right|$ is the number of observation points in partial sensing data $\mathcal{D}_n^{(m)}$.

Each computing node can support multiple subtasks. The computing demand in CPU cycles for all the subtasks placed at computing node n, denoted by μ_n, is given by

$$\mu_n = \sum_{m \in \mathcal{M}} e_n^{(m)} \mu^{(m)}, \quad \forall n \in \mathcal{N}^+. \tag{3.6}$$

Let $\boldsymbol{\alpha} = \{\alpha_n, \forall n \in \mathcal{N}^+\}$ be a continuous decision vector in $\mathbb{R}^{N+1}$, where α_n represents the fraction of computing resource usage at computing node n. Then, the computing time for all the subtasks placed at computing node n, denoted by t_n, is given by

$$t_n = \begin{cases} \frac{\mu_n}{\alpha_n f_n}, & \forall n \in \mathcal{N}^+ \text{ if } \alpha_n > 0 \\ 0, & \forall n \in \mathcal{N}^+ \text{ if } \alpha_n = 0. \end{cases} \tag{3.7}$$

3.6 Communication Model

Here, we focus on the communication model in the object classification phase, with the consideration of large raw data size for transmission in object classification.

If AV n provides partial sensing data $\mathcal{D}_n^{(m)}$ for subtask m, i.e, $s_n^{(m)} = 1$, and subtask m is placed at computing node $n' \in \mathcal{N}^+ \backslash \{n\}$, i.e., $e_{n'}^{(m)} = 1$, partial sensing data $\mathcal{D}_n^{(m)}$ should be transmitted from AV n to computing node n'. Let φ denote the data size (in bit) of one observation point in sensing data. Then, the total size of the sensing data transmitted over the communication link from AV n to computing node n', denoted by $\rho_{n,n'}$, is given by

$$\rho_{n,n'} = \varphi \sum_{m \in \mathcal{M}} s_n^{(m)} e_{n'}^{(m)} \left| \mathcal{D}_n^{(m)} \right|, \quad \forall n \in \mathcal{N},\ \forall n' \in \mathcal{N}^+ \backslash \{n\} \tag{3.8}$$

which incorporates all the partial sensing data transmitted between AV n and computing node n' for different subtasks. Note that the wireless communication link between AV n and computing node n' corresponds to a V2I link if $n' = N$, and corresponds to a V2V link otherwise. Orthogonal frequency division multiplexing (OFDM) based V2X transmissions are employed for AVs to communicate with each other and with the RSU. Consider that all the V2I and V2V links share a spectrum with total bandwidth B.

Let $\boldsymbol{\beta} = \{\beta_{n,n'}, \forall n \in \mathcal{N}, \forall n' \in \mathcal{N}^+\}$ denote a continuous bandwidth allocation decision matrix in $\mathbb{R}^{N \times (N+1)}$, where $\beta_{n,n'}$ represents the fraction of bandwidth allocated to the communication link from AV n to computing node n'. We have $\beta_{n,n} \equiv 0$ for $n \in \mathcal{N}$. The total fraction of

bandwidth allocated to different communication links should not exceed 1, given by

$$\sum_{n\in\mathcal{N}}\sum_{n'\in\mathcal{N}^+}\beta_{n,n'}\leq 1. \tag{3.9}$$

The average transmission rate over the communication link from AV n to computing node n', denoted by $R_{n,n'}$, is given by

$$R_{n,n'}=\beta_{n,n'}B\log_2\left(1+\frac{P_n\left|h_{n,n'}\right|^2 d_{n,n'}^{-\gamma}}{\sigma^2}\right) \tag{3.10}$$

where P_n denotes the uplink transmit power of AV n, $h_{n,n'}$ is the channel fading coefficient from AV n to computing node n', $d_{n,n'}$ is the distance between AV n and computing node n', γ is the path loss exponent, and σ^2 represents the noise power. Here, we assume constant distance $d_{n,n'}$ during the perception task period, with the consideration of low latency requirement for the perception task, e.g., $T=20ms$. For example, for two vehicles with a relative speed of $30\ km/h$ driving in the same direction on an urban road, the distance between the two vehicles changes for only around $17cm$ over a time duration of $20ms$, which is negligible [39]. Then, the transmission time for all the sensing data transmitted from AV n to computing node n', denoted by $t_{n,n'}$, is given by

$$t_{n,n'}=\begin{cases}\frac{\rho_{n,n'}}{R_{n,n'}}, & \forall n\in\mathcal{N},\forall n'\in\mathcal{N}^+\backslash\{n\}\text{ if }\beta_{n,n'}>0\\ 0, & \forall n\in\mathcal{N},\forall n'\in\mathcal{N}^+\text{ if }\beta_{n,n'}=0.\end{cases} \tag{3.11}$$

3.7 Summary

The chapter describes the system models covering main aspects of an autonoumous driving scenario including object spatial distributions, sensing data of AVs, a perception task of a target AV, a communoication and computing model.

Chapter 4

Problem Formulation and Solution

4.1 DNN-Based Classification Accuracy Estimation

For subtask m, the CNN classification accuracy (i.e., the estimated true class probability by the CNN classification model) fully depends on fused sensing data $\mathcal{D}^{(m)}$ and the CNN classification model. Consider a pre-trained CNN classification model with fixed parameters. Then, the main factor affecting the classification accuracy for subtask m is the data quality of fused sensing data $\mathcal{D}^{(m)}$, which is characterized by fused data quality vector $\boldsymbol{Z}^{(m)}$. As the fused data quality vector for each subtask depends on the corresponding data selection decision according to (3.3), the classification accuracy for each subtask also depends on the data selection decision.

The ground-truth CNN classification accuracy for subtask m, i.e., a_m, is unknown until fused sensing data $\mathcal{D}^{(m)}$ is processed by the CNN classification model. Hence, for data selection decision with accuracy satisfaction, accuracy estimation is required, to estimate the CNN classification accuracy for any candidate data selection decision given the partial sensing data quality

of each AV for each subtask. We profile an accuracy estimation function by a DNN model, referred to as DNN accuracy model, with fused data quality vector and cuboid region dimensions as inputs and classification accuracy as output, which is represented as

$$\hat{a}_m = \mathsf{f}^{DNN}\left(\boldsymbol{Z}^{(m)}, l_m^{(x)}, l_m^{(y)}, l_m^{(z)}\right), \quad \forall m \in \mathcal{M} \tag{4.1}$$

where $\hat{a}_m$ denotes the estimated CNN classification accuracy for subtask m, $\boldsymbol{Z}^{(m)}$ is the fused data quality vector for subtask m given by (3.3), and $\{l_m^{(x)}, l_m^{(y)}, l_m^{(z)}\}$ are cuboid region dimensions for object m. As an object with smaller size tends to require less observation points for accurate classification, the relationship between accuracy and the fused data quality vector is different for different classes of objects with different dimensions, which motivates us to include the cuboid region dimensions in the accuracy estimation function. The DNN model for accuracy estimation has an input dimension of $(K^3 + 3)$ and an output dimension of one, which can be pre-trained offline with simulated point cloud data in autonomous driving scenarios and refined online with real-world point cloud data. More details on DNN model training are given in Section 5.

The CNN classification model and the DNN accuracy model are related to each other. The CNN classification model takes the fused sensing data for a subtask as input and estimates a class probability vector as output; while the DNN accuracy model takes the fused data quality vector and cuboid region dimensions for a subtask as input and estimates a classification accuracy as output. For subtask m, the DNN accuracy model outputs an estimated CNN classification accuracy, $\hat{a}_m$, as an estimation for the ground-truth CNN classification accuracy, a_m, which can be obtained by processing fused sensing data $\mathcal{D}^{(m)}$ using the CNN classification model.

4.2 Problem Formulation

We focus on the object classification phase for a perception task initiated by the target AV. The assisting AVs can cooperate with the target AV for the perception task in terms of both cooperative sensing and cooperative computation. If assisting AV n provides partial sensing data for at least one subtask, i.e., $\sum_{m\in\mathcal{M}} s_n^{(m)} \geq 1$, we say that it participates in the cooperative sensing with the target AV. If at least one subtask is placed at computing node $n \in \mathcal{N}\backslash\{0\}$, i.e., $\sum_{m\in\mathcal{M}} e_n^{(m)} \geq 1$, assisting AV n participates in the cooperative computation with the target AV. The RSU with computing capability provided by a co-located edge server can also participate in cooperative computation by supporting the computation of at least one subtask.

To satisfy the accuracy and delay requirements of the perception task, we investigate a cooperative perception strategy which coordinates the differentiated sensing data selection and placement for multiple subtasks of the perception task, with efficient transmission resource allocation among multiple AVs and RSU, and with minimum computing resource consumption at all computing nodes. The decision variables for cooperative perception include binary data selection decision matrix, $\boldsymbol{s} \in \mathbb{R}^{N\times M}$, binary subtask placement decision matrix, $\boldsymbol{e} \in \mathbb{R}^{(N+1)\times M}$, continuous computing resource usage decision vector, $\boldsymbol{\alpha} \in \mathbb{R}^{N+1}$, and continuous bandwidth allocation decision matrix, $\boldsymbol{\beta} \in \mathbb{R}^{N\times(N+1)}$. The goal is to minimize the total transmission and computing resource consumption, while satisfying the quality-of-service (QoS) requirement of the perception task in terms of accuracy and delay.

A controller placed at the edge server makes the cooperative perception decision. Fig. 4.1 shows the interactions between the controller and AVs for cooperative perception. The relationship between object detection and classification phases is also illustrated. Specifically, with the

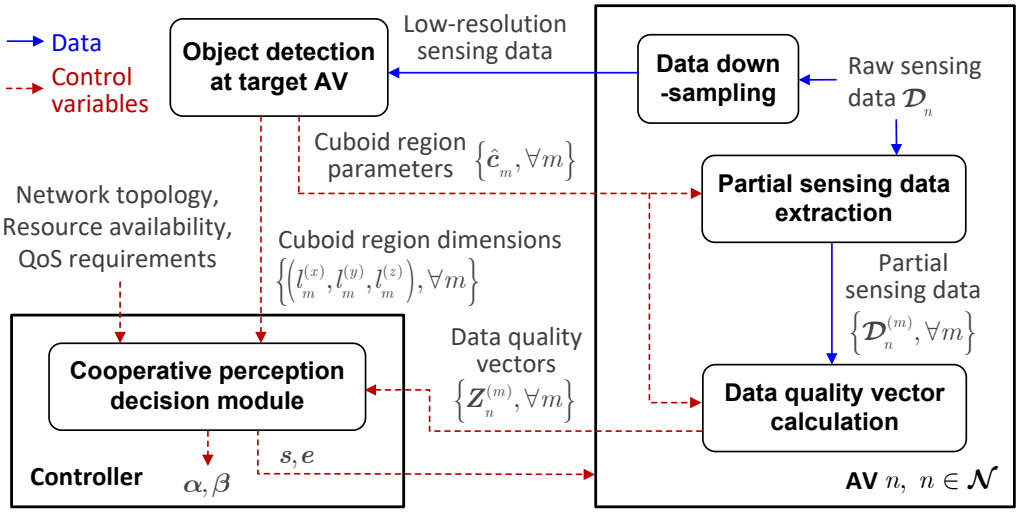


Figure 4.1: An illustration of interactions between controller and AVs for cooperative perception.

low-resolution sensing data of each AV, an object detection algorithm at the target AV estimates a set of cuboid region parameters for different objects, i.e., $\{\hat{\boldsymbol{c}}_m, \forall m \in \mathcal{M}\}$, based on which each AV can extract the partial sensing data for each object and calculate the corresponding data quality vectors. The controller takes the data quality vectors of different AVs and the cuboid region dimensions among the cuboid region parameters as input for cooperative perception decision. The data selection and subtask placement decisions are sent to corresponding AVs and computing nodes for execution of CNN classification.

According to Section 4.1, the CNN classification accuracy for subtask m, $\hat{a}_m$, can be estimated by a pre-trained DNN accuracy model, for each candidate data selection decision. For accuracy satisfaction, the accuracy of each subtask should satisfy the accuracy requirement A, given by

$$\mathsf{f}^{DNN}\left(\sum_{n\in\mathcal{N}} s_n^{(m)} \boldsymbol{Z}_n^{(m)}, l_m^{(x)}, l_m^{(y)}, l_m^{(z)}\right) \geq A, \quad \forall m \in \mathcal{M} \tag{4.2}$$

according to (3.3) and (4.1). In (4.2), $\boldsymbol{Z}_n^{(m)}$, $l_m^{(x)}$, $l_m^{(y)}$, $l_m^{(z)}$ are known parameters for cooperative perception decision, as indicated in Fig. 4.1.

Assume that all the AVs that are selected to send sensing data to other computing nodes start the data transmission simultaneously. We also assume that the computation at a computing node is started once the node receives selected sensing data for all the assigned subtasks. Then, for any subtask placed at computing node n', the subtask is completed in at most a time of $(\max_{n' \in \mathcal{N}} t_{n,n'}) + t_{n'}$. As the completion time for all the subtasks placed at different computing nodes should not violate the task delay requirement, T, we have a delay constraint for the perception task in object classification phase as

$$\max_{n' \in \mathcal{N}^+} \left(\left(\max_{n \in \mathcal{N}} t_{n,n'} \right) + t_{n'} \right) \leq T \tag{4.3}$$

which is equivalent to

$$t_{n,n'} + t_{n'} \leq T, \quad \forall n \in \mathcal{N},\ \forall n' \in \mathcal{N}^+. \tag{4.4}$$

Let $\boldsymbol{\chi} = \{\chi_{n,n'}, \forall n \in \mathcal{N}, \forall n' \in \mathcal{N}^+\}$ be an auxiliary binary decision matrix in $\mathbb{R}^{N \times (N+1)}$, with $\chi_{n,n'} = 1$ indicating that the wireless communication (either V2I or V2V) link between AV n and computing node n' is activated, i.e, AV n sends its partial sensing data for at least one subtask to computing node n', and $\chi_{n,n'} = 0$ otherwise. We have $\chi_{n,n} \equiv 0$ for $n \in \mathcal{N}$. The relationship among $\boldsymbol{\chi}$, data selection decision $\boldsymbol{s}$, and subtask placement decision $\boldsymbol{e}$ is given by

$$\frac{\sum_{m \in \mathcal{M}} s_n^{(m)} e_{n'}^{(m)}}{M} \leq \chi_{n,n'} \leq \sum_{m \in \mathcal{M}} s_n^{(m)} e_{n'}^{(m)}, \quad \forall n \in \mathcal{N},\ \forall n' \in \mathcal{N}^+. \tag{4.5}$$

According to (4.5), we have $\chi_{n,n'} = 0$ for $\sum_{m\in\mathcal{M}} s_n^{(m)} e_{n'}^{(m)} = 0$ when no sensing data are sent from AV n to computing node n' for any subtask, and $\chi_{n,n'} = 1$ otherwise. Under the assumption that each AV is equipped with one half-duplex transceiver radio, at most one communication link starting or ending at AV n can be activated, given by

$$\sum_{n'\in\mathcal{N}^+\setminus\{n\}} \chi_{n,n'} + \sum_{n'\in\mathcal{N}\setminus\{n\}} \chi_{n',n} \leq 1, \quad \forall n \in \mathcal{N}. \tag{4.6}$$

Specifically, an AV can participate in either V2I or V2V communication during one perception task period but not in both. For V2I communication, it acts as a transmitter. For V2V communication, it acts as either a transmitter or a receiver in at most one V2V pair. For example, if AV n is selected to provide sensing data for multiple subtasks, all the subtasks should be placed at either itself or another single computing node, as at most one V2I or V2V link from AV n to another computing node can be activated. If AV n is selected as the computing node for multiple subtasks, the corresponding partial sensing data for the subtasks can be sent from at most one AV other than AV n via a V2V link to AV n. Note that (4.6) does not hold for $n = N$, as multiple V2I links from different AVs to computing node N (i.e., the RSU) can be activated simultaneously due to multiple transceiver radios at the RSU.

Let o denote the total resource consumption cost for cooperative perception, which is a weighted summation of the total transmission and computing resource consumption, given by

$$o = \omega B \sum_{n\in\mathcal{N}} \sum_{n'\in\mathcal{N}^+} \beta_{n,n'} + (1-\omega) \sum_{n\in\mathcal{N}^+} \alpha_n f_n \tag{4.7}$$

with weighting factor $\omega \in (0, 1)$. To minimize the total resource consumption cost, o, for cooper-

ative perception in the autonomous driving scenario, we formulate a joint data selection, subtask placement, and resource allocation problem as optimization problem $\mathbf{P}1$, given by

$$
\begin{aligned}
\mathbf{P}1: \quad & \min_{\boldsymbol{e},\boldsymbol{s},\boldsymbol{\beta},\boldsymbol{\alpha},\boldsymbol{\chi}} \quad o \\
& \text{s.t.} \quad (3.4), (3.9), (4.2), (4.4), (4.5), (4.6) \\
& \qquad s_n^{(m)} \in \{0,1\}, \quad \forall m \in \mathcal{M},\ \forall n \in \mathcal{N} \\
& \qquad e_n^{(m)} \in \{0,1\}, \quad \forall m \in \mathcal{M},\ \forall n \in \mathcal{N}^+ \\
& \qquad \chi_{n,n'} \in \{0,1\}, \quad \forall n \in \mathcal{N},\ \forall n' \in \mathcal{N}^+ \\
& \qquad \chi_{n,n} = 0, \quad \forall n \in \mathcal{N} \\
& \qquad 0 \leq \beta_{n,n'} \leq 1, \quad \forall n \in \mathcal{N},\ \forall n' \in \mathcal{N}^+ \\
& \qquad \beta_{n,n} = 0, \quad \forall n \in \mathcal{N} \\
& \qquad 0 \leq \alpha_n \leq 1, \quad \forall n \in \mathcal{N}^+.
\end{aligned}
$$

Problem $\mathbf{P}1$ has constraints in terms of topology, accuracy, delay, and resource capacity. The topology constraints include the single computing node placement constraint for each subtask in (3.4), and the half-duplex communication constraints of each AV in (4.5) and (4.6). The accuracy and delay constraints are given by (4.2) and (4.4), respectively. Constraint (3.9) corresponds to the transmission resource capacity constraint, and the remaining constraints are range requirements for the decision variables.

4.3 Problem Solution

Problem $\mathbf{P}1$ is non-convex due to the nonlinear accuracy and delay constraints in (4.2) and (4.4). Moreover, the accuracy constraint in (4.2) which incorporates DNN-based accuracy estimation does not have a closed-form expression, making the problem intractable. We notice that the accuracy and topology constraints depend on only the binary decision variables including data selection and subtask placement, while the delay and resource capacity constraints depend on the joint decisions of data selection, subtask placement, and resource allocation. Once the data selection and subtask placement decisions are given, the delay and resource capacity constraints and the objective function depend on only the resource allocation decision.

We propose an iterative solution to problem $\mathbf{P}1$. In the solution, an outer module iteratively optimizes the data selection and subtask placement based on a genetic algorithm, which relies on an inner module to check the feasibility and cost. In each interaction between the inner and outer modules, the outer module provides the inner module with a candidate data selection and subtask placement solution, denoted by $(\boldsymbol{s}, \boldsymbol{e})$, which is feasible in terms of the accuracy and topology constraints. Given an $(\boldsymbol{s}, \boldsymbol{e})$ pair, the inner module optimizes the resource allocation for a minimal total resource consumption cost with delay constraint satisfaction, by solving a resource allocation subproblem. If the subproblem is infeasible, the accuracy and topology feasible candidate data selection and subtask placement solution, $(\boldsymbol{s}, \boldsymbol{e})$, is infeasible in terms of delay and resource constraints, and the outer module should abort the bad $(\boldsymbol{s}, \boldsymbol{e})$ pair. Otherwise, $(\boldsymbol{s}, \boldsymbol{e})$ is feasible in terms of all constraints in problem $\mathbf{P}1$, and a cost corresponding to $(\boldsymbol{s}, \boldsymbol{e})$ provided by the inner module should be evaluated by the outer module.

Fig. 4.2 illustrates the interactions between the outer module based on a genetic algorithm

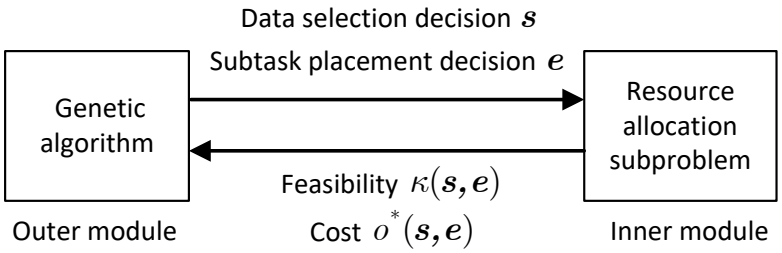


Figure 4.2: An illustration of interactions between genetic algorithm and resource allocation subproblem.

and the inner module which solves a resource allocation subproblem, where $\kappa\left(\boldsymbol{s},\boldsymbol{e}\right)$ denotes the feasibility of the resource allocation subproblem given $(\boldsymbol{s},\boldsymbol{e})$, and $o^*\left(\boldsymbol{s},\boldsymbol{e}\right)$ denotes the minimal total resource consumption cost obtained by solving the resource allocation subproblem given $(\boldsymbol{s},\boldsymbol{e})$, as defined in Subsection 4.3.1.

4.3.1 Resource Allocation Subproblem

Given an $(\boldsymbol{s},\boldsymbol{e})$ pair, the auxiliary binary link activation decision matrix, $\boldsymbol{\chi} \in \mathbb{R}^{N\times(N+1)}$, is uniquely determined according to (4.5). Based on known link activation status, $\boldsymbol{\chi}$, given $(\boldsymbol{s},\boldsymbol{e})$, we consider a network topology represented as a directed graph $G = \{\mathcal{N}^{\mathtt{A}},\mathcal{L}^{\mathtt{A}}\}$, where $\mathcal{N}^{\mathtt{A}} \subset \mathcal{N}^{+}$ is a set of nodes composed of the starting and ending nodes (either AV or RSU) of all the activated links, and $\mathcal{L}^{\mathtt{A}}$ is a set of directed links composed of all the activated links. For nodes $n \in \mathcal{N}$ and $n' \in \mathcal{N}^{+}$, let (n,n') denote a directed link from node n to node n'. We have $n \in \mathcal{N}^{\mathtt{A}}$, $n' \in \mathcal{N}^{\mathtt{A}}$ and $(n,n') \in \mathcal{L}^{\mathtt{A}}$ if link (n,n') is activated, i.e., $\chi_{n,n'} = 1$.

Let $\boldsymbol{\alpha}^{\mathtt{A}} = \{\alpha_n, \forall n \in \mathcal{N}^{\mathtt{A}}\}$ denote the fractions of computing resource usage at the nodes in set $\mathcal{N}^{\mathtt{A}}$, and let $\boldsymbol{\beta}^{\mathtt{A}} = \{\beta_{n,n'}, \forall (n,n') \in \mathcal{L}^{\mathtt{A}}\}$ denote the fractions of bandwidth allocated to the links in set $\mathcal{L}^{\mathtt{A}}$. Given the data selection and subtask placement decisions, i.e., $(\boldsymbol{s},\boldsymbol{e})$, the remaining unknown resource allocation decision variables in problem $\mathbf{P}1$ include $\boldsymbol{\alpha}^{\mathtt{A}}$ and $\boldsymbol{\beta}^{\mathtt{A}}$.

Other resource allocation decision variables are all equal to 0.

The computing demand in CPU cycles at node $n \in \mathcal{N}^{\mathtt{A}}$, i.e., μ_n, and the total size of the sensing data transmitted over link $(n, n') \in \mathcal{L}^{\mathtt{A}}$, i.e., $\rho_{n,n'}$, are constant parameters given an $(\boldsymbol{s}, \boldsymbol{e})$ pair, according to (3.6) and (3.8). Let constant $C_n = \frac{\mu_n}{f_n}$ denote the computing time at node n using all the available computing resources, i.e., $\alpha_n = 1$. Let constant $C_{n,n'} = \frac{\rho_{n,n'}}{B \log_2\left(1+P_n|h_{n,n'}|^2 d_{n,n'}^{-\gamma} \big/ \sigma^2\right)}$ denote the transmission time over link (n, n') using whole bandwidth B, i.e., $\beta_{n,n'} = 1$.

The resource allocation subproblem for the inner module is to minimize the total resource consumption cost by determining the resource allocation decision variables, $\boldsymbol{\alpha}^{\mathtt{A}}$ and $\boldsymbol{\beta}^{\mathtt{A}}$, while satisfying the task delay requirement under resource capacity constraints. The resource allocation subproblem is formulated as an optimization problem, given by

$$
\begin{aligned}
\mathbf{P}2: \quad \min_{\boldsymbol{\alpha}^{\mathtt{A}}, \boldsymbol{\beta}^{\mathtt{A}}} \quad & \omega B \sum_{(n,n') \in \mathcal{L}^{\mathtt{A}}} \beta_{n,n'} + (1-\omega) \sum_{n \in \mathcal{N}^{\mathtt{A}}} \alpha_n f_n \\
\text{s.t.} \quad & \sum_{(n,n') \in \mathcal{L}^{\mathtt{A}}} \beta_{n,n'} \leq 1 \\
& \frac{C_{n,n'}}{\beta_{n,n'}} + \frac{C_{n'}}{\alpha_{n'}} \leq T, \quad \forall (n, n') \in \mathcal{L}^{\mathtt{A}} \\
& 0 < \beta_{n,n'} \leq 1, \quad \forall (n, n') \in \mathcal{L}^{\mathtt{A}} \\
& 0 < \alpha_n \leq 1, \quad \forall n \in \mathcal{N}^{\mathtt{A}}
\end{aligned}
\tag{4.8}
$$

where the delay constraint in (4.8) is rewritten based on (3.7), (3.11) and (4.4). Let $\kappa(\boldsymbol{s}, \boldsymbol{e})$ be a binary feasibility flag for problem $\mathbf{P}2$ given $(\boldsymbol{s}, \boldsymbol{e})$, with $\kappa(\boldsymbol{s}, \boldsymbol{e}) = 1$ if problem $\mathbf{P}2$ is feasible given $(\boldsymbol{s}, \boldsymbol{e})$, and $\kappa(\boldsymbol{s}, \boldsymbol{e}) = 0$ otherwise. Let $o^*(\boldsymbol{s}, \boldsymbol{e})$ be the optimal objective value of problem $\mathbf{P}2$ given $(\boldsymbol{s}, \boldsymbol{e})$, corresponding to the minimal total resource consumption cost with optimal

resource allocation. Note that $o^*(\boldsymbol{s}, \boldsymbol{e})$ is defined only when problem $\mathbf{P}2$ is feasible given $(\boldsymbol{s}, \boldsymbol{e})$, i.e., $\kappa(\boldsymbol{s}, \boldsymbol{e}) = 1$. When problem $\mathbf{P}2$ is infeasible given $(\boldsymbol{s}, \boldsymbol{e})$, $o^*(\boldsymbol{s}, \boldsymbol{e})$ is undefined.

Due to the nonlinear delay constraint in (4.8), problem $\mathbf{P}2$ is non-convex. We transform the problem to a second-order cone programming (SOCP) problem with zero optimality gap by introducing several auxiliary decision variables, with details presented in Appendix.

4.3.2 Genetic Algorithm

The outer module in the solution to problem $\mathbf{P}1$ jointly optimizes the data selection decision, $\boldsymbol{s}$, and the subtask placement decision, $\boldsymbol{e}$, by solving an optimization problem given by

$$
\begin{aligned}
\mathbf{P}3: \quad & \min_{\boldsymbol{e},\boldsymbol{s},\boldsymbol{\chi}} \quad o^*(\boldsymbol{s}, \boldsymbol{e}) \\
& \text{s.t.} \quad (3.4), (4.2), (4.5), (4.6) \\
& \qquad \kappa(\boldsymbol{s}, \boldsymbol{e}) = 1 \\
& \qquad s_n^{(m)} \in \{0, 1\}, \quad \forall m \in \mathcal{M},\ \forall n \in \mathcal{N} \\
& \qquad e_n^{(m)} \in \{0, 1\}, \quad \forall m \in \mathcal{M},\ \forall n \in \mathcal{N}^+ \\
& \qquad \chi_{n,n'} \in \{0, 1\}, \quad \forall n \in \mathcal{N},\ \forall n' \in \mathcal{N}^+ \\
& \qquad \chi_{n,n} = 0, \quad \forall n \in \mathcal{N}
\end{aligned}
\tag{4.9}
$$

where $o^*(\boldsymbol{s}, \boldsymbol{e})$ and $\kappa(\boldsymbol{s}, \boldsymbol{e})$ are the minimal total resource consumption cost and the binary feasibility flag of resource allocation subproblem $\mathbf{P}2$ for a given $(\boldsymbol{s}, \boldsymbol{e})$ pair, respectively, both provided by the inner module given $(\boldsymbol{s}, \boldsymbol{e})$. Let $\varsigma(\boldsymbol{s}, \boldsymbol{e})$ be a binary feasibility flag for problem $\mathbf{P}3$ given a joint data selection and subtask placement solution, $(\boldsymbol{s}, \boldsymbol{e})$, with $\varsigma(\boldsymbol{s}, \boldsymbol{e}) = 1$ indicating

that all the constraints in problem $\mathbf{P}3$ are satisfied for the given $(\boldsymbol{s}, \boldsymbol{e})$ pair, and $\varsigma(\boldsymbol{s}, \boldsymbol{e}) = 0$ otherwise. Specifically, to check whether the accuracy constraint in (4.2) is satisfied or not for a given $(\boldsymbol{s}, \boldsymbol{e})$ pair, the DNN model for accuracy estimation should run M times to estimate the accuracy for M subtasks, given the data selection decision, $\boldsymbol{s}$, and other known parameters including data quality vectors and cuboid region dimensions. To check whether constraint (4.9) is satisfied or not for a given $(\boldsymbol{s}, \boldsymbol{e})$ pair, the feasibility of resource allocation subproblem Problem $\mathbf{P}2$ should be checked given the $(\boldsymbol{s}, \boldsymbol{e})$ pair.

To solve problem $\mathbf{P}3$ with the intractable DNN based accuracy constraint in (4.2), we propose a genetic algorithm (GA) which iteratively optimizes the joint data selection and subtask placement decision by selecting the candidate solutions (referred to as individuals in GA) with a lower cost (referred to better fitness in GA) in a current generation to reproduce new individuals in the next generation [14, 41, 47]. Here, an individual corresponds to a candidate joint data selection and subtask placement solution, $(\boldsymbol{s}, \boldsymbol{e})$.

Let integer τ be the generation index. A population of J individuals in generation $\tau + 1$ are reproduced based on the population in generation τ. Let $\mathcal{J} = \{0, \cdots, J-1\}$ denote the index set for J individuals in each generation. An individual in GA is usually represented by a sequence of genes. Here, we consider M genes for each individual, each gene corresponding to the data selection and subtask placement decision for one subtask. Specifically, the m-th gene of an individual corresponding to a candidate solution $(\boldsymbol{s}, \boldsymbol{e})$, denoted by $\boldsymbol{v}_m$, is a concatenated data selection and subtask placement decision vector for subtask m, given by

$$\boldsymbol{v}_m = [\boldsymbol{s}_m, \boldsymbol{e}_m], \quad \forall m \in \mathcal{M} \tag{4.10}$$

where $\boldsymbol{s}_m = \{s_n^{(m)}, \forall n \in \mathcal{N}\}$ and $\boldsymbol{e}_m = \{e_n^{(m)}, \forall n \in \mathcal{N}^+\}$ represent the data selection and subtask placement decisions for subtask m, respectively. Let $\boldsymbol{v}_m^{\tau,j}$ denote the m-th gene of the j-th individual in generation τ. Then, the j-th individual in generation τ containing a set of M genes, denoted by $\mathbf{V}^{\tau,j}$, is represented as

$$\mathbf{V}^{\tau,j} = \left\{\boldsymbol{v}_m^{\tau,j},\ \forall m \in \mathcal{M}\right\}. \tag{4.11}$$

Individual $\mathbf{V}^{\tau,j}$ corresponds to the j-th candidate joint data selection and subtask placement solution in generation τ, denoted by $(\boldsymbol{s}^{\tau,j}, \boldsymbol{e}^{\tau,j})$. Then, the cost of individual $\mathbf{V}^{\tau,j}$, denoted by $o^{\tau,j}$, is given by $o^{\tau,j} = o^*\left(\boldsymbol{s}^{\tau,j}, \boldsymbol{e}^{\tau,j}\right)$, which is the minimal total resource consumption cost obtained by solving problem $\mathbf{P}2$ given $(\boldsymbol{s}^{\tau,j}, \boldsymbol{e}^{\tau,j})$. The feasibility of individual $\mathbf{V}^{\tau,j}$, denoted by $\varsigma^{\tau,j}$, is given by $\varsigma^{\tau,j} = \varsigma\left(\boldsymbol{s}^{\tau,j}, \boldsymbol{e}^{\tau,j}\right)$, which represents the feasibility of problem $\mathbf{P}3$ for $(\boldsymbol{s}^{\tau,j}, \boldsymbol{e}^{\tau,j})$.

Each individual in generation τ has a different probability to be selected as a parent to reproduce offspring individuals in generation $\tau + 1$. Let $p^{\tau,j}$ denote the selection probability for individual $\mathbf{V}^{\tau,j}$ in generation τ, given by

$$p^{\tau,j} = 1 - \frac{o^{\tau,j}}{\sum_{j'=0}^{J-1} o_{j'}^{\tau}} \tag{4.12}$$

which indicates that the individuals with a lower cost among a population have a higher probability to be selected for reproduction.

Let $\mathbf{\Phi}^{\tau}$ denote the population in generation τ, which is a set of individuals with size J, represented as

$$\mathbf{\Phi}^{\tau} = \left\{\mathbf{V}^{\tau,j},\ \forall j \in \mathcal{J}\right\}. \tag{4.13}$$

There are two key operations in GA, i.e., crossover and mutation, to reproduce new candidate individuals for the next generation based on the population in a current generation, as follows.

- **Crossover:** For crossover, two individuals in the current population are randomly selected based on the cost-dependent selection probability in (4.12). Let $\mathbf{V}^1 = \{\boldsymbol{v}_m^1, \forall m \in \mathcal{M}\}$ and $\mathbf{V}^2 = \{\boldsymbol{v}_m^2, \forall m \in \mathcal{M}\}$ denote the two selected individuals. Then, based on a random gene position, $\hat{m} \in \mathcal{M}$, a new candidate individual with combined genes from both individuals, denoted by $\mathbf{V}^C$, is generated by inheriting the genes of individual $\mathbf{V}^1$ before the random gene position and the genes of individual $\mathbf{V}^2$ after it, given by

$$\mathbf{V}^C = \left\{\boldsymbol{v}_m^1, 0 \leq m \ \leq \hat{m} - 1\right\} \cup \left\{\boldsymbol{v}_m^2, \hat{m} \leq m \ \leq M - 1\right\}; \tag{4.14}$$

- **Mutation:** For mutation, one individual in the current population is randomly selected. Then, based on a random gene position, a new candidate individual is generated by replacing the gene at the randomly selected position of the original individual by a random gene.

The GA algorithm for joint data selection and subtask placement operates as follows, with a pseudo code presented in Algorithm 1. It starts with a population of J randomly generated feasible individuals in the initial generation, denoted by $\mathbf{\Phi}^0$, among which the j-th individual, $\mathbf{V}^{0,j} \in \mathbf{\Phi}^0$, has cost $o^{0,j}$ and feasibility $\varsigma^{0,j} = 1$. Then, the GA algorithm iteratively reproduces a new population including J individuals with a potentially lower average cost for each new generation, until a maximum number of iterations, Γ, is reached.

Specifically, in iteration τ, the J new individuals forming population $\mathbf{\Phi}^{\tau+1}$ are reproduced based on population $\mathbf{\Phi}^{\tau}$. Let $\mathbf{V}^{\tau,j^*}$ denote an elite individual in population $\mathbf{\Phi}^{\tau}$ with the minimal

Algorithm 1: Genetic algorithm for data selection and subtask placement

Initialization: Generate an initial population $\mathbf{\Phi}^0 = \{\mathbf{V}^{0,j}, \forall j \in \mathcal{J}\}$ containing J random feasible individuals for generation $\tau = 0$;
for *each generation with* $0 \leq \tau \leq \Gamma - 1$ **do**
  The elite individual with the minimal cost in population $\mathbf{\Phi}^\tau$ survives to generation $\tau + 1$, i.e., $\mathbf{V}^{\tau+1,0} = \mathbf{V}^{\tau,j^*}$ and $o^{\tau+1,0} = o^{\tau,j^*}$;
  for $2 \leq j \leq J$ **do**
    Selection: Randomly select two individuals $\mathbf{V}^1$ and $\mathbf{V}^2$ with cost o^1 and o^2 from population $\mathbf{\Phi}^\tau$ based on the cost-dependent selection probability in (4.12);
    Set $\mathbf{V}^{\tau+1,j} = \mathbf{V}^1$ and $o^{\tau+1,j} = o^1$ by default;
    Generate candidate individual $\hat{\mathbf{V}}$ by **crossover** between $\mathbf{V}^1$ and $\mathbf{V}^2$ with a probability of p_C, and by **mutation** based on $\mathbf{V}^1$ with a probability of p_M;
    if *Candidate individual* $\hat{\mathbf{V}}$ *is feasible (i.e.,* $\hat{\varsigma} = 1$*)* **then**
      Set $\mathbf{V}^{\tau+1,j} = \hat{\mathbf{V}}$ and $o^{\tau+1,j} = \hat{o}$.

cost, where j^* is the index of the elite individual, given by

$$j^* = \arg\min_{j \in \mathcal{J}} \ o^{\tau,j}. \tag{4.15}$$

The elite individual, $\mathbf{V}^{\tau,j^*}$, survives to generation $\tau + 1$, and becomes an individual with index $j = 0$ in population $\mathbf{\Phi}^{\tau+1}$, i.e., $\mathbf{V}^{\tau+1,0} = \mathbf{V}^{\tau,j^*}$. To generate the j-th individual for $1 \leq j \leq J-1$ in generation $\tau + 1$, i.e., $\mathbf{V}^{\tau+1,j}$, two individuals in population $\mathbf{\Phi}^\tau$, denoted as $\mathbf{V}^1$ and $\mathbf{V}^2$, are randomly selected based on the cost-dependent selection probability. Let o^1 and o^2 denote the cost of $\mathbf{V}^1$ and $\mathbf{V}^2$, respectively. A candidate individual, $\hat{\mathbf{V}}$, is generated by crossover between $\mathbf{V}^1$ and $\mathbf{V}^2$ with a probability of p_C, and by mutation based on one of the selected individuals, e.g., $\mathbf{V}^1$, with a probability of p_M. Let $\hat{\varsigma}$ and $\hat{o}$ denote the feasibility and cost of candidate individual $\hat{\mathbf{V}}$. Note that $\hat{o}$ is defined only if candidate individual $\hat{\mathbf{V}}$ is feasible. If candidate individual $\hat{\mathbf{V}}$ is feasible, i.e., $\hat{\varsigma} = 1$, $\mathbf{V}^{\tau+1,j}$ is set as $\hat{\mathbf{V}}$, with $o^{\tau+1,j} = \hat{o}$. Otherwise, $\mathbf{V}^{\tau+1,j}$ is set

as one of the selected individuals, e.g., $\mathbf{V}^1$, by default, with $o^{\tau+1,j} = o^1$.

4.4 Summary

In the chapter, we first deign a DNN accuracy model to estimate the classification accuracy for each subtask depending on the corresponding data selection decision. Then, we formulate a problem for joint data selection, subtask placement and resource allocation. The solution for the formulated problem is finally given.

Chapter 5

Performance Evaluation

5.1 Simulation Setup

As the genetic algorithm relies on a DNN accuracy model for accuracy satisfaction according to (4.2), we should pre-train a DNN model for accuracy estimation, with an input dimension of (K^3+3) and an output dimension of one, where K has four candidate values in $\{1,2,3,4\}$. The default value of K is set to 3. The DNN model has two hidden layers with $(32,16)$ neurons between the input and output layers. The activation function for each hidden layer is `Relu`. Due to the K-dependent input dimension, the DNN model for accuracy estimation has a K-dependent model structure. To train the DNN model, we first create a training dataset consisting of 5600 labeled training data. Specifically, we use an automated driving toolbox in Matlab and simulate multiple random autonomous driving scenarios with a random number of AVs and objects randomly distributed on a road. Each AV is equipped with one sensor or depth camera on top of the vehicle. For each simulated scenario, a number of simulated point clouds are generated at the

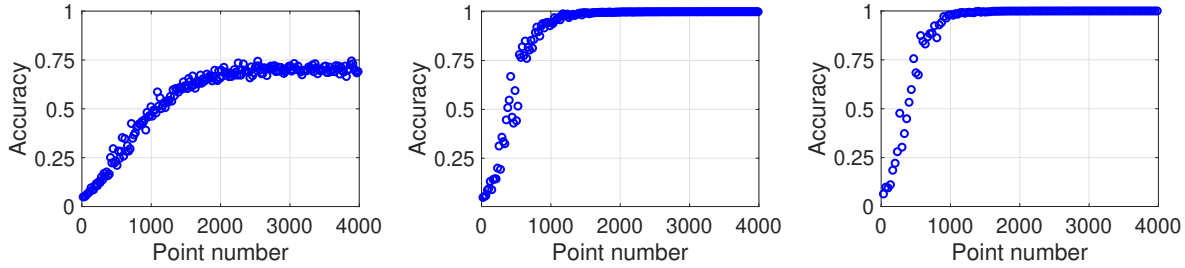
AVs, which are randomly fused to generate more simulated LiDAR point clouds with different point number and spatial distribution. Based on the random simulated point clouds, we obtain multiple random object-level sensing data segments by extracting the partial sensing data for each object from each point cloud. Each random object-level sensing data segment corresponds to one K-dependent training data, which is a $(K^3 + 3)$-dimension vector concatenated from a K^3-dimension data quality vector for the sensing data segment and three dimension values (including length, width, height) for the cuboid region of the corresponding object. Note that the cuboid region parameters are known for each class of objects with given size. To obtain the label associated with one training data, the corresponding sensing data segment is processed using a CNN classification model, which estimates a class probability vector with a certain accuracy. The ground-truth CNN classification accuracy, i.e., the estimated probability for the true class, is set as the label associated with the corresponding training data. Here, we use a well-known VoxelNet model developed for 3D point cloud inputs as the CNN classification model [63]. With the training dataset, we can train the DNN model for accuracy estimation by minimizing the mean squared error (MSE) between the estimated and ground-truth CNN classification accuracy for each training data in the training dataset.

With the pre-trained DNN model for accuracy estimation, we can evaluate the performance of the proposed cooperative sensing and computation scheme. We consider a VEC-enabled autonomous driving scenario with one target AV and three assisting AVs, on a unidirectional road segment with a length of $50\ m$ under the coverage of one RSU, as illustrated in Fig. 3.1. There are six objects distributed in the target AV's RoI, including two trucks, two cars, one pedestrian and one cyclist. The perception task of the target AV is to correctly detect and classify the six objects in the RoI. We set the perception delay requirement as $T = 20$ ms, and set the accuracy

Table 5.1: System parameters in simulation

Parameters	Value
Bandwidth (B)	20 MHz
Noise power (σ^2)	10^{-13} W
Transmit power (P_n)	1 W
Path-loss exponent (γ)	3.4
Channel fading coefficient ($\|h_{n,n'}\|^2$)	1
Available computing resources at AV n (f_n)	10^{10} cycle/s
Available computing resources at RSU (f_N)	2×10^{11} cycle/s
Data size per observation point (φ)	192 bit

requirement, A, among two candidate values in $\{0.7, 0.9\}$. The system parameters are given in Table 5.1. For simplicity, we assume identical noise power, transmit power, channel fading coefficient, and computing capability among all the AVs. We use the automated driving toolbox to simulate the 3D point cloud generation by both the target and assisting AVs in the considered transportation scenario. To obtain the low-resolution sensing data at each AV for object detection, the simulated raw sensing data (i.e., point cloud) at each AV is down-sampled by averaging the observation points in each $0.1 \times 0.1 \times 0.1\ m^3$ grid to one point in the low-resolution sensing data. Then, the cuboid region parameters for each object are estimated based an existing object detection algorithm. Specifically, we use the Euclidean clustering and L-shape fitting algorithm for object detection [58, 64]. Based on the cuboid region parameters, the partial sensing data for each object is extracted from the simulated raw sensing data at each AV, and the corresponding data quality vectors are calculated. Given the cuboid region dimensions and data quality vectors, the genetic algorithm for joint cooperative sensing and computation is executed, which gives a joint data selection, subtask placement, and resource allocation solution for the perception task



(a) Using data from target AV (b) Using data from one assisting AV (c) Using fused data from both AVs

Figure 5.1: Relationship between classification accuracy and point number for three groups of sensing data segments for a truck.

in the considered transportation scenario. For each feasible candidate joint data selection and subtask placement decision in the iterations of the genetic algorithm, the optimal resource allocation is obtained by solving an SOCP problem $\mathbf{P}4$ in Appendix using the Gurobi optimization solver [1]. The computation intensity (in cycle/point), ϵ, is selected among four candidate values from 10000 to 40000. The default computation intensity is set to $\epsilon = 30000$ cycle/point. By evaluating the total communication resource consumption in the unit of MHz and the total computing resource consumption in the unit of Giga cycles/s, the two terms in (4.7) are comparable. We set weighting factor ω in (4.7) to 0.33.

5.2 Simulation Results

We first evaluate the impact of point number and spatial distribution of the observation points for an object on the object classification accuracy. We use the sensing data of the target AV and one assisting AV for a truck in front of the target AV for the evaluation, as illustrated in Fig. 3.4. The assisting AV is at the front left corner of the truck. Three raw sensing data segments for the truck are used, corresponding to the raw partial sensing data of the target AV, the raw partial sensing

data of the assisting AV, and the fused raw partial sensing data of both AVs, which contain a total number of 5610, 12594 and 18204 observation points respectively. For each raw sensing data segment, a group of sensing data segments with different data resolution are generated by down-sampling the raw sensing data segment using different down-sampling ratios from 0.01 to 1. The sensing data segments within one group contain a different number of observation points for the truck. The three groups of sensing data segments with different data resolution are processed using the CNN classification model, and a classification accuracy is obtained for each sensing data segment. Fig. 5.1 shows the relationship between classification accuracy and point number for the three groups of sensing data segments. As the observation points of the target AV for the truck are concentrated at the back side of the truck with a low sensing data diversity, we see in Fig. 5.1 (a) that the classification accuracy increases slowly to between 70% and 75% as the point number increases and gradually saturates with further increase of point number, inferring that adding more observation points without improving the sensing data quality brings limited accuracy gain, especially when the point number is already large. By contrast, as the observation points of the assisting AV for the truck mainly spread over the front and left sides, providing more sensing data diversity, we see in Fig. 5.1 (b) that the classification accuracy increases more rapidly with the increase of point number and approaches 100% with less than 1500 observation points, inferring that less observation points with better sensing data diversity are required to achieve the same classification accuracy. However, the accuracy gain brought by further increasing the sensing data diversity is also limited, as demonstrated in Fig. 5.1 (c). Although the fused sensing data broadly covers three sides (left, front and back sides) of the truck, the relationship between classification accuracy and point number is similar to that in Fig. 5.1 (b). With the same number of observation points covering two or three sides of the

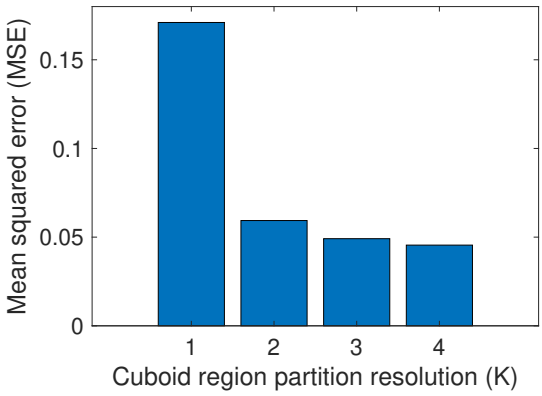


Figure 5.2: Performance of the K-dependent DNN model for accuracy estimation in terms of MSE.

truck, a similar classification accuracy is achieved, although the fused data covering three sides has a better sensing data diversity. It infers that the sensing data diversity by covering the left and front sides of the truck is sufficient for accurate classification.

For each candidate value of cuboid region partition resolution K among $\{1, 2, 3, 4\}$, a DNN model for accuracy estimation with a K-dependent input dimension is trained based on a K-dependent training dataset. The training error of each DNN model is evaluated by the MSE between the estimated classification accuracy values and the ground-truth classification accuracy labels. Fig. 5.2 shows the relationship between K and the MSE of each K-dependent DNN model. We observe that a DNN model corresponding to a larger K has a smaller MSE, indicating a better performance in estimating the CNN classification accuracy. The reason is that a data quality vector (as a part of DNN model input) with a larger cuboid region partition resolution captures the spatial distribution of observation points in more details. For example, when K is equal to 1, the data quality vector is reduced to a scalar which represents the total number of observation points in a sensing data segment, losing the spatial distribution information. As a result, the MSE at $k = 1$ is large. When K is increased to 2, the data quality vector has $K^3 = 8$

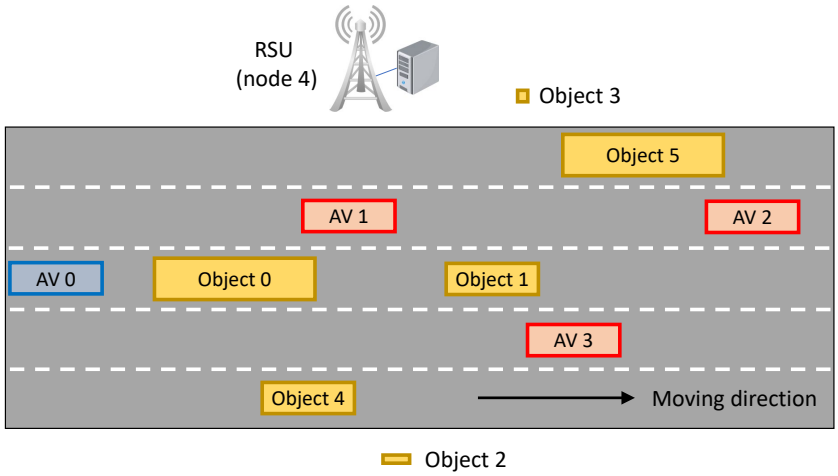
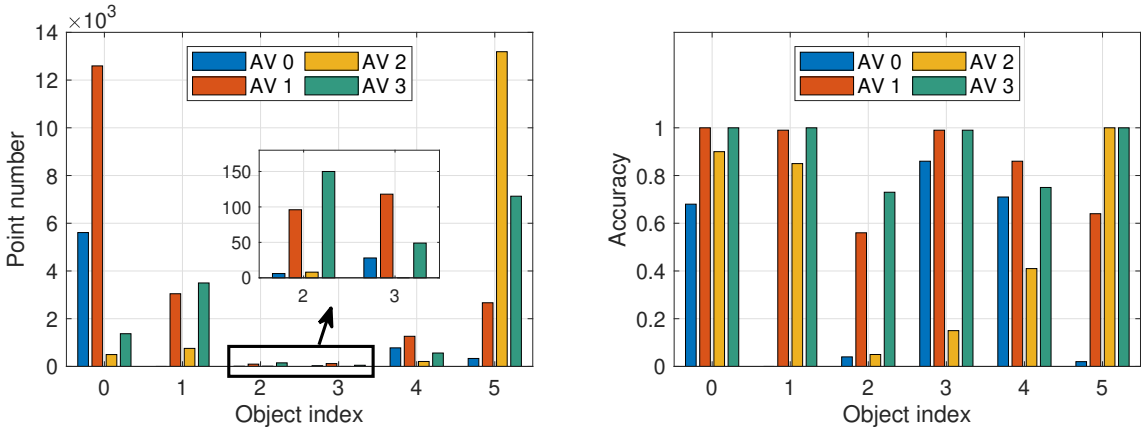


Figure 5.3: A 2D illustration of the considered VEC-enabled autonomous driving scenario.

dimensions, which can coarsely capture the spatial distribution of observation points. Hence, a big drop is observed in the MSE. However, the improvement in MSE becomes less significant by further increasing K. The potential reason is that there are too many zeros in the K^3-dimension data quality vector when K is large, which does not contribute to significantly more information gain in charactering the spatial distribution of observation points, especially when the observation points for an object are concentrated in a small subset of sub-regions inside the 3D cuboid region for the object.

To evaluate the benefit of cooperative sensing in term of accuracy improvement, we first evaluate the object classification accuracy without cooperative sensing for each object. In this case, the sensing data of a single AV is used for the classification of all the objects, without data fusion with the sensing data from other AVs. Fig. 5.3 shows a 2D version for the considered VEC-enabled autonomous driving scenario in Fig. 3.1, where the indexes of AVs, RSU, and objects are indicated. Specifically, AV 0 corresponds to the target AV, and other AVs are all assisting AVs. AV n ($0 \leq n \leq 3$) is also referred to as node n when it serves as a computing



(a) Point number

(b) Accuracy

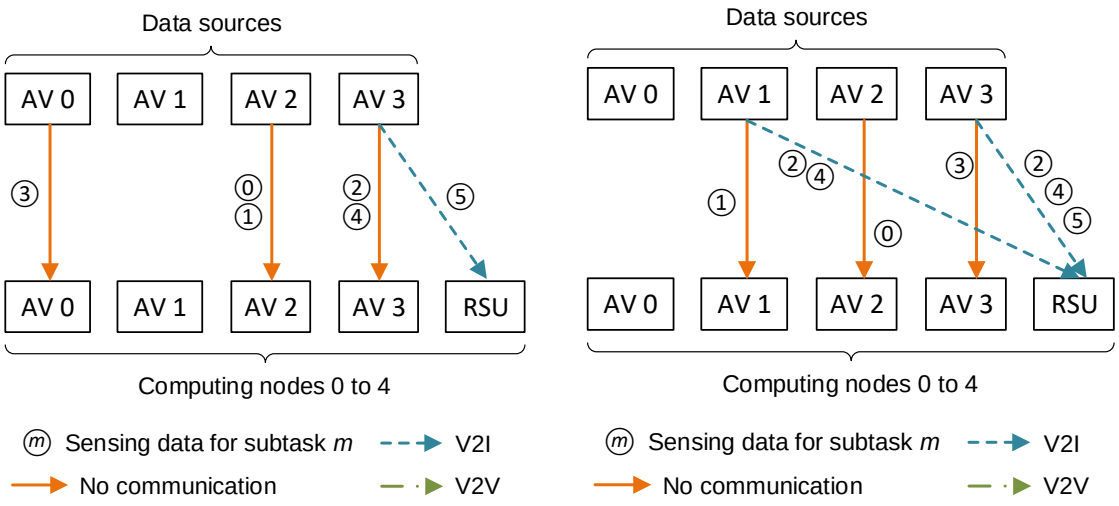
Figure 5.4: Object classification performance based on single AV's sensing data.

node. The RSU is indicated as node 4. Objects 0 and 5 corresponding to two trucks have a larger size, objects 1 and 4 corresponding to two cars have a medium size, and objects 2 and 3 corresponding to a cyclist and a pedestrian respectively are small. Fig. 5.4 shows the number of observation points at each AV for each object, and the corresponding object classification accuracy by processing each single AV's sensing data for each object. We discuss the results for object 0 as an example. For object 0, each AV has line-of-sight observations, and the number of observation points decreases with increasing distance between the AV and object 0. Both AV 1 and AV 3 provide a good view for object 0, with a sufficient number of observation points for an accuracy of 1. Although AV 2 is farther from object 0 and has much less observation points covering the front left sides, it still provides a good accuracy at around 0.9. However, due to the short distance between AV 0 and object 0, AV 0 only sees the back side of object 0 with very limited sensing data diversity, leading to a low accuracy below 0.7. Due to better sensing data diversity at AV 2 and AV 3 for object 0, the accuracy provided by both AVs is better than that of AV 0 while using much less observation points.

Fig. 5.4 shows that none of the AVs can achieve a classification accuracy beyond 0.7 for all the objects by purely relying on its own sensing data, demonstrating the necessity for cooperative sensing under the simulation settings. Let (n, m) denote an AV-object pair with AV n and object m, whose accuracy is evaluated by processing the partial sensing data of AV n for object m. The AV-object pairs with a poor accuracy below 0.7 are affected by different factors, such as blockage, distance and limited sensing diversity. By observing the spatial relationships among AVs and objects in Fig. 5.3, we find that the AV-object pairs with poor accuracy due to blockage include $(0, 1)$, $(0, 2)$, $(2, 2)$, $(2, 3)$ and $(0, 5)$, and the AV-object pairs with poor accuracy due to long distance include $(1, 2)$ and $(1, 5)$. AV-object pair $(0, 0)$ has a poor accuracy due to limited sensing data diversity, and AV-object pair $(2, 4)$ has a poor accuracy due to both long distance and partial sensor occlusion. We also observe that less observation points are required to achieve a similar accuracy for an object with a smaller size, which is demonstrated by the accuracy of AV-object pairs $(1, 1)$ and $(1, 3)$ as an example.

Impact of Accuracy Requirement

The proposed cooperative sensing and computation strategy allows differentiated data selection and subtask placement for each subtask, referred to as differentiated cooperation (DC) strategy. Fig. 5.5 illustrates the data selection and subtask placement solutions by the DC strategy for two different accuracy requirements at $A = 0.7$ and $A = 0.9$ respectively. The default value for computation intensity is used, i.e., $\epsilon = 30000$ cycle/point. An arrow starting from AV n to computing node n', denoted by (n, n'), indicates that the sensing data of AV n is used for at least one subtask placed at computing node n'. The numbers in circles beside arrow (n, n') indicate the indexes of subtasks with sensing data selection from AV n and placement at computing node



(a) $A = 0.7$ (b) $A = 0.9$

Figure 5.5: Data selection and subtask placement solutions for two different accuracy requirements.

n'. If $n = n'$, no communication is required for the selected sensing data, as the sensing data are processed at the AV where it is generated. If $n \neq n'$, the arrow indicates V2V communication for $n' < 4$ or V2I communication for $n' = 4$.

For a low accuracy requirement at $A = 0.7$, no cooperative sensing is required, as illustrated in Fig. 5.5 (a). Each subtask uses the corresponding partial sensing data from a single AV, without data fusion with other AVs. Most subtasks except subtask 5 consume no communication resources, by processing the selected sensing data locally. Each of them is placed at an AV whose local sensing data can satisfy the accuracy requirement, $A = 0.7$, with the minimum computing demand. For example, for object 0, AV 2 has the minimum point number among three AVs whose sensing data can satisfy the accuracy requirement, as shown in Fig. 5.4. Thus, subtask 0 selects sensing data from AV 2 and processes the data locally at AV 2, which incurs no communication cost, as shown in Fig. 5.5 (a). For subtasks 1 to 4, we also observe similar

data selection and subtask placement strategies. Subtask 5 is the only subtask placed at the RSU and with communication demand. Both AV 2 and AV 3 have sensing data with accuracy satisfaction for subtask 5. However, neither of them can process the sensing data for subtask 5 locally without violating the delay requirement. Hence, subtask 5 must be offloaded to the RSU with more computing resources for delay improvement. For example, the local computing time in processing 7134 observation points for object 5 at AV 3 with all the available computing resources is 21.4 ms, which violates the delay requirement, $T = 20$ ms. As the sensing data for object 5 at AV 3 has a smaller size than that at AV 2, the sensing data of AV 3 is selected for subtask 5 to reduce the communication and computing demand.

However, for a high accuracy requirement at $A = 0.9$, cooperative sensing is required for subtasks 2 and 4, as neither of them can rely on sensing data from a single AV for accuracy satisfaction, as illustrated in Fig. 5.4 and Fig. 5.5 (b). AV 1 and AV 3 are selected to provide sensing data for both subtasks 2 and 4 by the proposed DC strategy. Due to the half-duplex communication constraints, subtasks 2 and 4 must be placed at the same computing node, which can be AV 1, AV 3, or the RSU. They cannot be placed at an AV other than AV 1 or AV 3, as another AV cannot receive data from both AV 1 and AV 3 at the same time. As the local sensing data at AV 1 can satisfy the accuracy requirement with the minimum computing demand for subtask 1, AV 1 is the preferred computing node for subtask 1. Similarly, AV 3 is the preferred computing node for subtask 3. Due to the limited local computing resources, subtasks 2 and 4 cannot be co-located with subtask 1 or subtask 3. Hence, subtasks 2 and 4 are placed at the RSU, and the corresponding selected sensing data are transmitted to the RSU via V2I communication. The data selection and subtask placement for both subtasks 0 and 5 remain unchanged when increasing the accuracy requirement from 0.7 to 0.9, as shown in Fig. 5.5.

Table 5.2: Resource consumption using DC and UC strategies for the different accuracy requirements

Cooperation strategy	Link bandwidth (MHz)		Computing resources (Giga cycles/s)	
	$A = 0.7$	$A = 0.9$	$A = 0.7$	$A = 0.9$
Differentiated cooperation (DC)	1.94	2.39	24.56	31.74
Unified cooperation (UC)	9.76	9.76	98.59	98.59

We compare the performance of the proposed DC strategy with a unified cooperation (UC) benchmark, where the sensing data for any subtask is selected from a same group of AVs. As each AV in the group should provide its sensing data for different subtasks, all the subtasks should be placed at the same computing node due to the half-duplex communication constraints. We evaluate the resource efficiency of both the DC and UC strategies for two different accuracy requirements, $A = 0.7$ and $A = 0.9$, with the default computation intensity, i.e., $\epsilon = 30000$ cycle/point. Table 5.2 shows the total transmission and computing resource consumption for both strategies. For the DC strategy, we observe an increase in both the total transmission and computing resource consumption when increasing the accuracy requirement from 0.7 to 0.9. Generally, more observation points tend to be required for a better accuracy, which transforms to an increase in both the communication and computing resource consumption. If the accuracy requirement becomes more stringent, AVs which provide sensing data with more observation points are selected. Subtask 1 and subtask 3 are such examples in our simulation scenario, when the accuracy requirement is increased from 0.7 to 0.9, as shown in Fig. 5.5. Moreover, a subtask which originally has single AV's sensing data for a lower accuracy may require more sensing data from different AVs for data fusion, to achieve a higher accuracy. Subtask 2 and subtask 4

are such examples in our simulation scenario.

As illustrated in Fig. 5.4, cooperative sensing is necessary to provide an accuracy of 0.7 and above, as no single AV can provide sensing data with good quality for all subtasks. Hence, at least two AVs should cooperate to provide sensing data for the subtasks. With the UC strategy, AV 1 and AV 3 are selected to provide sensing data for all the subtasks, which are placed at the RSU, for an accuracy requirement of $A = 0.7$. Specifically, the sensing data of AV 1 is sufficient for subtasks 0, 1, and 3 to achieve an accuracy beyond 0.7, and the sensing data of AV 3 is sufficient for subtasks 0, 1, 3, and 5 to achieve an accuracy beyond 0.7. For subtask 4, a fusion of the sensing data from both AVs can help to meet the accuracy requirement. However, due to the unified cooperation, the corresponding partial sensing data of both AVs are selected for each subtask by the UC strategy, which is resource inefficient. Due to the large computing demand in processing the fused partial sensing data from both AVs for each subtask, the subtasks are placed at the RSU with more computing resources for delay satisfaction, which also increases the communication resource consumption via V2I communication. By processing the redundant sensing data at the RSU, the accuracy for the six subtasks are 1, 0.99, 0.91, 1, 0.96, and 1, respectively, which are over-provisioned for both accuracy requirements at $A = 0.7$ and $A = 0.9$. Hence, we see a huge gap between the total transmission and computing resource consumption by the DC and UC strategies in Table 5.2, demonstrating the resource efficiency of the proposed DC strategy.

Impact of Computation Intensity

The computation intensity, ϵ, which captures the computing resource demand for each point, depends on the specific object classification model [49]. As the proposed cooperative sensing

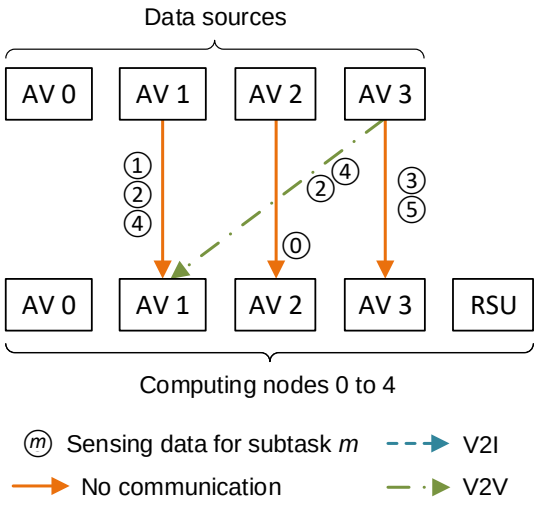


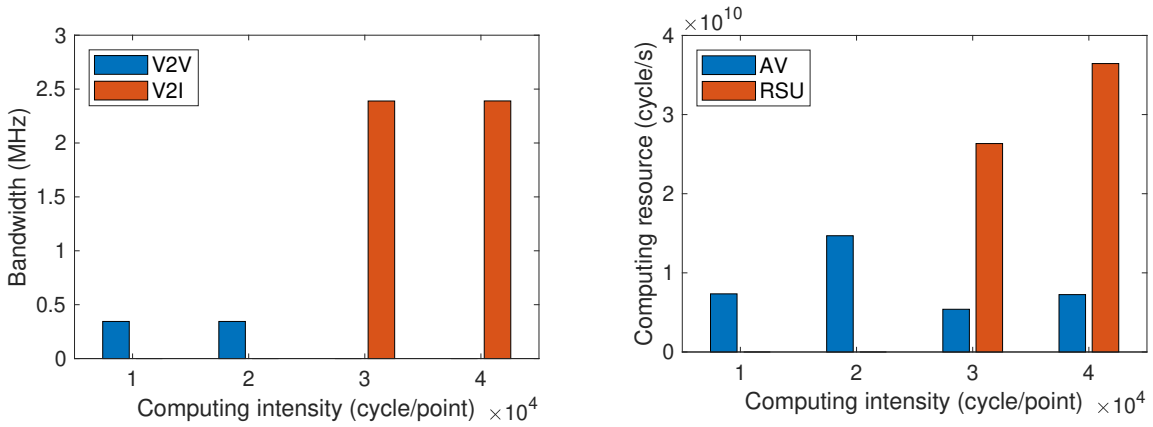
Figure 5.6: Data selection and subtask placement solution for $A = 0.9$ at $\epsilon = 10000, 20000$.

and computation scheme can be generalized for other object classification models with different computation intensity, we evaluate the impact of computation intensity, ϵ, on the performance of the proposed DC strategy, and then compare the performance between the DC and UC strategies, with the increase of ϵ from 10000 to 40000 cycle/point. Here, we set the accuracy requirement as $A = 0.9$.

By varying ϵ from 10000 to 40000, we find two different data selection and subtask placement solutions by the DC strategy. Specifically, the solutions for a low computing intensity at $\epsilon = 10000, 20000$ are the same, as illustrated in Fig. 5.6, and the solutions for a high computing intensity at $\epsilon = 30000, 40000$ are also the same, as illustrated in Fig. 5.5 (b). As the accuracy requirement is fixed at $A = 0.9$, the data selection solution keeps unchanged with the increase of ϵ, as the accuracy depends only on data selection decision and some constant parameters such as data quality vectors. Accordingly, the total point number and sensing data size for each subtask is unchanged with the increase of ϵ. However, the computation intensity affects the subtask

placement solution, as the computing resource demand for each subtask increases linearly with computation intensity. With a low computation intensity at $\epsilon = 10000, 20000$, no subtask is offloaded to the RSU for delay satisfaction. All the subtasks are placed at AVs for communication resource efficiency, as illustrated in Fig. 5.6. Specifically, for both subtask 2 and subtask 4, the corresponding partial sensing data from AV 1 requires no communication, as both subtasks are placed at AV 1, and the corresponding partial sensing data from AV 3 is transmitted to AV 1 via a V2V link. In comparison, with a high computation intensity at $\epsilon = 30000, 40000$, three subtasks including subtasks 2, 4, and 5 are offloaded to the RSU for delay satisfaction, as illustrated in Fig. 5.5 (b). Accordingly, the corresponding selected sensing data are transmitted to the RSU via V2I links, requiring more bandwidth resources, as shown in Fig. 5.7 (a). We also observe the same bandwidth consumption for different values of ϵ corresponding to the same data selection and subtask placement solution, as the sensing data size is the same for each subtask. Fig. 5.7 (b) shows the computing resource consumption at AVs and RSU by the DC strategy with the increase of ϵ. With a low computation intensity at $\epsilon = 10000, 20000$, there is no computing resource consumption at the RSU, and the computing resource consumption at AVs increases linearly with ϵ. With a high computation intensity at $\epsilon = 30000, 40000$, there is computing resource consumption at both AVs and RSU, and the computing resource consumption at either AVs or RSU increases linearly with ϵ.

Fig. 5.8 shows the resource consumption comparison between the DC and UC strategies as ϵ increases. The unified data selection solution by the UC strategy remains unchanged as ϵ increases from 10000 to 40000, in meeting the accuracy requirement of all subtasks. Specifically, the sensing data of AV 1 and AV 3 are selected by the UC strategy. Hence, the radio resource consumption remains unchanged for the UC strategy, as the same selected sensing data are trans-



(a) Communication resources

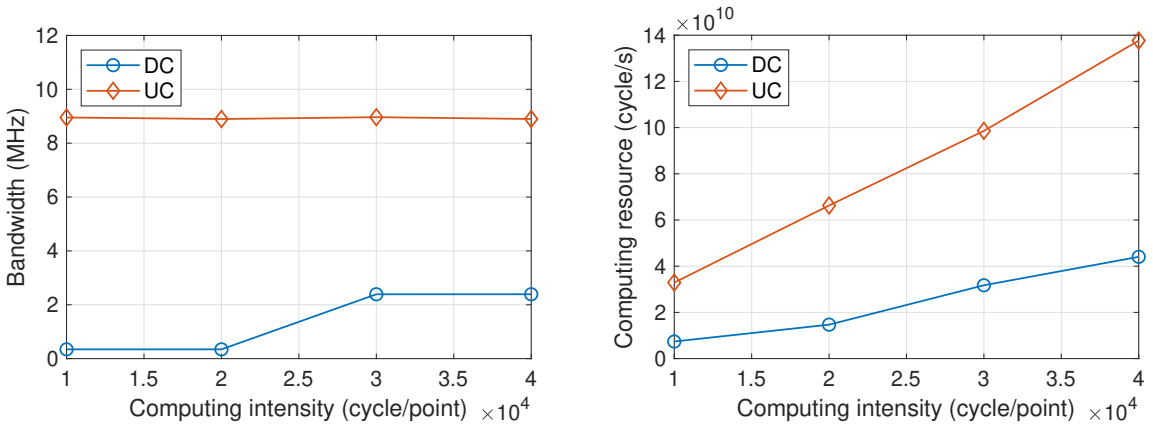
(b) Computing resources

Figure 5.7: Communication and computing resource consumption versus computation intensity (ϵ).

mitted over V2I links for the different values of ϵ. The unified placement for all the subtasks also remains unchanged at the RSU, leading to a linear increase in the total computing resource consumption by the UC strategy as ϵ increases. Due to the unified data selection and subtask placement, the total number of observation points in the selected sensing data by the UC strategy is large, resulting in a large computing demand even at low computation intensity. The performance of the DC strategy in Fig. 5.8 is consistent with the results shown in Fig. 5.7. We observe a significant improvement by the proposed DC strategy in both communication and computing resource efficiency as compared with the UC benchmark.

Impact of Object Number

Finally, we evaluate the impact of object number on the performance of the proposed DC strategy. The performance between the DC and UC strategies are compared when increasing the object number N from 1 to 6. Given the object number N, only above N objects from the considered object list (their spatial distributions are described in Fig. 3.1) are deployed in the



(a) Communication resources

(b) Computing resources

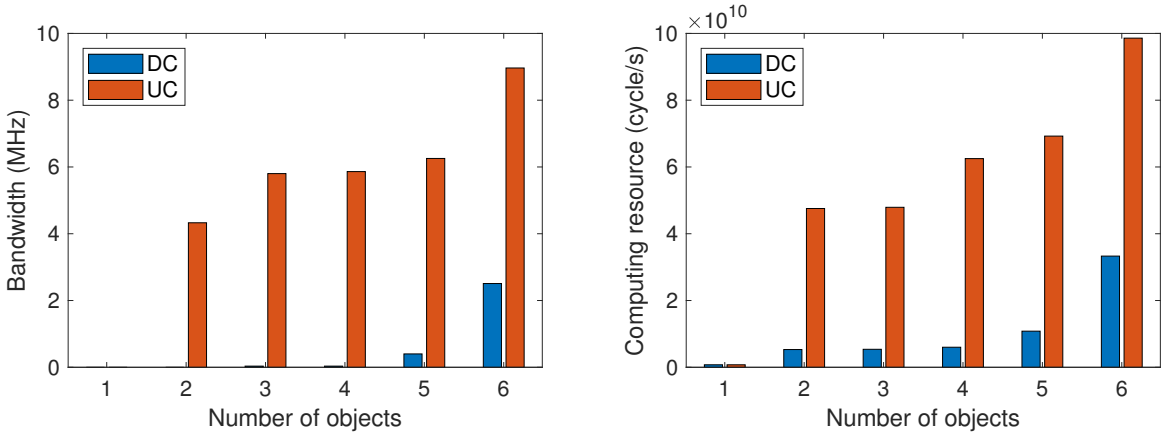
Figure 5.8: Performance comparison between the DC and UC strategies with the increase of ϵ for $A = 0.9$.

autonomous driving scenario, and the corresponding subtasks are needed to be executed for accurate object classification. Here, we set the accuracy requirement as $A = 0.9$ and the computing intensity as $\epsilon = 30000$ cycle/point. Fig. 5.9 shows the total transmission and computing resource consumption for both strategies.

For the DC strategy, we observe an increase in both total transmission and computing resource consumption when increasing the object number from 1 to 6. Generally, more objects require more observation points to satisfy the accuracy requirement, which further requires more communication and computing resource consumption. When the object number is not larger than 2, V2X communication is not required in the DC strategy as Subtask 1 and 2 can be separately processed by 2 AVs based on their own sensing data. When the object number becomes 3, the communication resource consumption increases as Subtask 3 requires more sensing data from different AVs for data fusion. When the object number increases to 4, no additional communication is required as Subtask 4 can be processed by an AV based on its sensing data. Therefore the communication resource consumption remains the same as that of 3 objects. As the object

number varies from 4 to 5, the communication resource consumption increases. This is because Subtask 5 relies on the fused sensing data from other AVs. Finally, when the object number increases to 6, an increase in comunication resource consumption is observed, bur It is not because Subtask 6 requires more sensing data from other AVs. Actually, the computing resources of AVs can support the execution of only the above 5 subtasks. Therefore, when Subtask 6 is required, some subtasks have to be offloaded to the RSU for delay satisfaction. Therefore, the corresponding selected sensing data have to be transmitted to the RSU via V2I links, requiring more bandwidth resources.

When comparing the performance between the DC and UC strategies, we first observe the computing and communication resource consumption of the DC strategy is the same as that of the UC strategy when the object number is 1. That is, these two strategies give the same data selection and subtask placement solution, where the task is processed by AV 2 based on its sensing data. When the object number is larger than 1, the advantage of the DC strategy in resource comsumption is observed. The reason is the object-level differentiated data selection scheme of the DC strategy effectively reduces the volume of redundant data. For example, when the object number is 2, the DC strategy separately selects AV 2 and AV 1 to provide sensing data for Subtasks 1 and 2 respectively. In constrast, the UC stratgy selects an unified data set for the task which inevitably increases the input size and computing demand of the perception task. Therefore, the proposed DC strategy can improve in both communication and computing resource efficiencies as compared with the UC benchmark.



(a) Communication resources

(b) Computing resources

Figure 5.9: Performance comparison between the DC and UC strategies for $A = 0.9$ and $\epsilon = 30000$.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

In this thesis, a cooperative perception problem on raw data level is studied in a VEC-enabled autonomous driving scenario, to satisfy the accuracy and delay requirements of a perception task initiated by a target AV for object detection and classification, with communication and computing resource efficiency. A cooperative sensing and computation strategy is investigated, which allows multiple assisting AVs to share their sensing data, and allows both the assisting AVs and the RSU to share their computing resources, both for the perception task. Specifically, the perception task is partitioned into multiple parallel subtasks, and a differentiated data selection and placement scheme among subtasks is proposed for cooperative sensing and computation. With the differentiated cooperation strategy, both the communication and computing resource efficiency is improved, by selecting a minimum amount of sensing data from different AVs for each subtask with accuracy satisfaction, and distributing the computation for different subtasks

among the RSU and multiple AVs in an efficient communication manner.

An optimization problem is formulated to determine the data selection and placement for different subtasks with optimal resource allocation, to minimize the total communication and computing resource consumption with delay and accuracy satisfaction. For accuracy requirement, a DNN-based classification accuracy model is proposed, to estimate the classification accuracy for a subtask based on the data selection decision and sensing data quality. A genetic algorithm based solution is proposed, which iteratively updates the data selection and subtask placement decision, based on the feasibility of a resource allocation subproblem and the minimal resource consumption cost with optimal resource allocation. Simulation results demonstrate the accuracy improvement with raw data level cooperative sensing, and the resource efficiency of the proposed differentiated cooperation strategy in comparison with a benchmark unified cooperation strategy.

6.2 Future Work

Although the proposed cooperative sensing and computing strategy meets the real-time processing requirement and achieves perception accuracy satisfaction with enhanced communication and computing resource efficiency, some research issues require further studies:

- The performance of object detection and classification should be maximized based on the sensing data quality to increase the resource efficiency of cooperative perception. In addition to the distance and angle between the sensor and object, we should take account of how sensing data quality affects the performance. For example, the quality of sensing data can be represented in terms of co-variances of distance and angle measurement errors from different AVs, in addition to their sensor hardware noises. Note that the co-variances in

general depend on the relative positions between the target AV and assistant AVs. Sensing data with larger measurement error co-variances should be weighted less in their contribution to the object detection and classification under consideration in this research.

- In this research, we investigate a cooperative sensing and computing strategy only for a perception task of a target AV. In practical autonomous driving scenarios, multiple AVs usually simultaneously require perception tasks in their corresponding RoIs. Therefore, it is necessary to investigate cooperative sensing and computing strategies for multiple perception tasks. Considering perception tasks from neighboring AVs often have partial overlapped RoIs, merging these tasks by running only one perception in overlapped regions and sharing the corresponding results to neighboring AVs allows for significant saving in communication and computing resources. However, this method constrains these tasks to the same choice of data selection and placement.

- In the proposed cooperative sensing strategy, each selected AV for an object classification subtask needs to provide all of its sensing data in the corresponding cuboid region. However, although only an AV is selected for a subtask, the subtask's accuracy can be over-provisioned due to providing more sensing data than required at the cost of more communication and computing resource consumption. Therefore, a potential solution to further improve resource efficiency is not only to select AVs, but also to determine parts of sensing data from the selected AVs in the corresponding cuboid regions for subtasks.

References

[1] Gurobi optimizer reference manual. http://www.gurobi.com, 2022. [Online; accessed 3-March-2022].

[2] Mohamed K Abdel-Aziz, Cristina Perfecto, Sumudu Samarakoon, Mehdi Bennis, and Walid Saad. Vehicular cooperative perception through action branching and federated reinforcement learning. *IEEE Trans. Commun.*, 70:891–903, 2022.

[3] Sekh Arif Ahmed, Debi Prosad Dogra, Samarjit Kar, Renuka Patnaik, Seung-Cheol Lee, Heeseung Choi, Gi Pyo Nam, and Ig-Jae Kim. Query-based video synopsis for intelligent traffic monitoring applications. *IEEE Trans. Intell. Transp. Syst.*, 21(8):3457–3468, 2019.

[4] Qier An and Yuan Shen. On the information coupling and propagation of visual 3D perception in vehicular networks with position uncertainty. *IEEE Trans. Veh. Technol.*, 70(12):13325–13339, 2021.

[5] Eduardo Arnold, Mehrdad Dianati, Robert de Temple, and Saber Fallah. Cooperative perception for 3D object detection in driving scenarios using infrastructure sensors. *IEEE Trans. Intell. Transp. Syst.*, 2020. to appear, doi: 10.1109/TITS.2020.3028424.

[6] Roberto Baldessari, D Scanferla, L Le, W Zhang, and A Festag. Joining forces for vanets: A combined transmit power and rate control algorithm. In *Proc. Int. WIT'10*, 2010.

[7] Luca Barbieri, Stefano Savazzi, Mattia Brambilla, and Monica Nicoli. Decentralized federated learning for extended sensing in 6G connected vehicles. *Veh. Commun.*, 33:100396, 2022.

[8] Diego Borsetti and Javier Gozálvez. Infrastructure-assisted geo-routing for cooperative vehicular networks. In *Proc. IEEE VNC'10*, pages 255–262, 2010.

[9] Mattia Brambilla, Monica Nicoli, Gloria Soatti, and Francesco Deflorio. Augmenting vehicle localization by cooperative sensing of the driving environment: Insight on data association in urban traffic scenarios. *IEEE Trans. Intell. Transp. Syst.*, 21(4):1646–1663, 2019.

[10] Qi Chen, Xu Ma, Sihai Tang, Jingda Guo, Qing Yang, and Song Fu. F-cooper: Feature based cooperative perception for autonomous vehicle edge computing system using 3D point clouds. In *Proc. ACM/IEEE SEC'19*, pages 88–100, 2019.

[11] Qi Chen, Sihai Tang, Qing Yang, and Song Fu. Cooper: Cooperative perception for connected autonomous vehicles based on 3d point clouds. In *Proc. IEEE Conf. ICDCS'19*, pages 514–524, 2019.

[12] Kai-Wei Chiang, Guang-Je Tsai, Hone-Jay Chu, and Naser El-Sheimy. Performance enhancement of INS/GNSS/Refreshed-SLAM integration for acceptable lane-level navigation accuracy. *IEEE Trans. Veh. Technol.*, 69(3):2463–2476, 2020.

[13] Bin Dai, Fanglin Xu, Yuanyuan Cao, and Yang Xu. Hybrid sensing data fusion of cooperative perception for autonomous driving with augmented vehicular reality. *IEEE Syst. J.*, 15(1):1413–1422, 2020.

[14] Shuiguang Deng, Longtao Huang, Javid Taheri, and Albert Y Zomaya. Computation offloading for service workflow in mobile cloud computing. *IEEE Trans. Parallel Distrib. Syst.*, 26(12):3317–3329, 2014.

[15] Michele Drigo, Wenhui Zhang, Roberto Baldessari, Long Le, Andreas Festag, and Michele Zorzi. Distributed rate control algorithm for vanets (drcv). In *Proc. ACM VANET'10*, pages 119–120, 2009.

[16] Mingjin Gao, Rujing Shen, Long Shi, Wen Qi, Jun Li, and Yonghui Li. Task partitioning and offloading in dnn-task enabled mobile edge computing networks. *IEEE Trans. Mob. Comput.*

[17] Li He, Guoliang Liu, Guohui Tian, Jianhua Zhang, and Ze Ji. Efficient multi-view multi-target tracking using a distributed camera network. *IEEE Sens. J.*, 20(4):2056–2063, 2019.

[18] Takamasa Higuchi, Marco Giordani, Andrea Zanella, Michele Zorzi, and Onur Altintas. Value-anticipating v2v communications for cooperative perception. In *Proc. IEEE IV'19*, pages 1947–1952, 2019.

[19] João Eduardo Hoffmann, Hilkija Gaïus Tosso, Max Mauro Dias Santos, João Francisco Justo, Asad Waqar Malik, and Anis Ur Rahman. Real-time adaptive object detection and tracking for autonomous vehicles. *IEEE Trans. Veh. Technol.*, 6(3):450–459, 2020.

[20] Md Hossain, Ibrahim Elshafiey, Abdulhameed Al-Sanie, et al. Cooperative vehicle positioning with multi-sensor data fusion and vehicular communications. *Wireless Netw.*, 25(3):1403–1413, 2019.

[21] Hui Huang, Huiyun Li, Cuiping Shao, Tianfu Sun, Wenqi Fang, and Shaobo Dang. Data redundancy mitigation in V2X based collective perceptions. *IEEE Access*, 8:13405–13418, 2020.

[22] Seong-Woo Kim, Baoxing Qin, Zhuang Jie Chong, Xiaotong Shen, Wei Liu, Marcelo H Ang, Emilio Frazzoli, and Daniela Rus. Multivehicle cooperative driving using cooperative perception: Design and experimental validation. *IEEE Trans. Intell. Transp. Syst.*, 16(2):663–680, 2014.

[23] Seongwoo Kim, Wei Liu, Marcelo H Ang, Emilio Frazzoli, and Daniela Rus. The impact of cooperative perception on decision making and planning of autonomous vehicles. *IEEE Trans. Intell. Transp. Syst.*, 7(3):39–50, 2015.

[24] Swarun Kumar, Lixin Shi, Nabeel Ahmed, Stephanie Gil, Dina Katabi, and Daniela Rus. Carspeak: a content-centric network for autonomous driving. *Proc. ACM SIGCOMM'12*, 42(4):259–270, 2012.

[25] Sampo Kuutti, Saber Fallah, Konstantinos Katsaros, Mehrdad Dianati, Francis Mccullough, and Alexandros Mouzakitis. A survey of the state-of-the-art localization techniques and their potentials for autonomous vehicle applications. *IEEE Internet Things J.*, 5(2):829–846, 2018.

[26] Jeongho Kwak, Yeongjin Kim, Joohyun Lee, and Song Chong. DREAM: Dynamic resource and task allocation for energy minimization in mobile cloud systems. *IEEE J. Select. Areas Commun.*, 33(12):2510–2523, 2015.

[27] Zongdian Li, Tao Yu, Ryuichi Fukatsu, Gia Khanh Tran, and Kei Sakaguchi. Towards safe automated driving: design of software-defined dynamic mmwave V2X networks and PoC implementation. *IEEE open J. Veh. Technol.*, 2:78–93, 2021.

[28] Adam Lipowski and Dorota Lipowska. Roulette-wheel selection via stochastic acceptance. *Physica A*, 391(6):2193–2196, 2012.

[29] Kai Liu, Joseph KY Ng, Victor CS Lee, Sang H Son, and Ivan Stojmenovic. Cooperative data scheduling in hybrid vehicular ad hoc networks: Vanet as a software defined network. *IEEE/ACM Tran. Netw.*, 24(3):1759–1773, 2015.

[30] Lei Liu, Ming Zhao, Miao Yu, Mian Ahmad Jan, Dapeng Lan, and Amirhosein Taherkordi. Mobility-aware multi-hop task offloading for autonomous driving in vehicular edge computing and networks. *IEEE Trans. Intell. Transp. Syst.*, 2022. to appear, doi: 10.1109/TITS.2022.3142566.

[31] Shaoshan Liu, Jie Tang, Zhe Zhang, and Jean-Luc Gaudiot. Computer architectures for autonomous driving. *Computer*, 50(8):18–25, 2017.

[32] Yi Liu, Huimin Yu, Shengli Xie, and Yan Zhang. Deep reinforcement learning for offloading and resource allocation in vehicle edge computing and networks. *IEEE Trans. Veh. Technol.*, 68(11):11158–11168, 2019.

[33] Daniel Maturana and Sebastian Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *Proc. IEEE/RSJ IROS'15*, pages 922–928, 2015.

[34] Jim Misener. Sae connected vehicle standards. *Proc. CES'16*, 2016, 2016.

[35] Dmitri Moltchanov, Andrey Samuylov, Ekaterina Lisovskaya, Roman Kovalchukov, Vyacheslav Begishev, Eduard Sopin, Yuliya Gaidamaka, and Yevgeni Koucheryavy. Performance characterization and traffic protection in street multi-band millimeter-wave and microwave deployments. *IEEE Trans. Commun.*, 21(1):163–178, 2021.

[36] Cristina Perfecto, Javier Del Ser, and Mehdi Bennis. Millimeter-wave V2V communications: Distributed association and beam alignment. *IEEE J. Select. Areas Commun.*, 35(9):2148–2162.

[37] Jin-Chun Piao and Shin-Dug Kim. Real-time visual-inertial SLAM based on adaptive keyframe selection for mobile AR applications. *IEEE Trans. Multimedia*, 21(11):2827–2836, 2019.

[38] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proc. IEEE CVPR'2017*, pages 652–660, 2017.

[39] Yanli Qi, Yiqing Zhou, Ya-Feng Liu, Ling Liu, and Zhengang Pan. Traffic-aware task offloading based on convergence of communication and sensing in vehicular edge computing. *IEEE Internet Things J.*, 8(24):17762–17777, 2021.

[40] Guanhua Qiao, Supeng Leng, Ke Zhang, and Yejun He. Collaborative task offloading in vehicular edge multi-access networks. *IEEE Commun. Mag.*, 56(8):48–54, 2018.

[41] Akhlaqur Rahman, Jiong Jin, Antonio L Cricenti, Ashfaqur Rahman, and Ambarish Kulkarni. Communication-aware cloud robotic task offloading with on-demand mobility for smart factory maintenance. *IEEE Trans. Ind. Electron.*, 15(5):2500–2511, 2018.

[42] Andreas Rauch, Felix Klanner, Ralph Rasshofer, and Klaus Dietmayer. Car2x-based perception in a high-level fusion architecture for cooperative perception systems. In *Proc. IEEE IV'12*, pages 270–275, 2012.

[43] Kei Sakaguchi, Ryuichi Fukatsu, Tao Yu, Eisuke Fukuda, Kim Mahler, Robert Heath, Takeo Fujii, Kazuaki Takahashi, Alexey Khoryaev, Satoshi Nagata, et al. Towards mmwave V2X in 5G and beyond to support automated driving. *IEICE Trans. Commun.*, 2020.

[44] Florian Seeliger and Klaus Dietmayer. Inter-vehicle information-fusion with shared perception information. In *Proc. IEEE ITSC'14*, pages 2087–2093, 2014.

[45] Jianbing Shen, Dajiang Yu, Leyao Deng, and Xingping Dong. Fast online tracking with detection refinement. *IEEE Trans. Intell. Transp. Syst.*, 19(1):162–173, 2017.

[46] Ibrahim Sorkhoh, Dariush Ebrahimi, Ribal Atallah, and Chadi Assi. Workload scheduling in vehicular networks with edge cloud capabilities. *IEEE Trans. Veh. Technol.*, 68(9):8472–8486, 2019.

[47] Fei Sun, Fen Hou, Nan Cheng, Miao Wang, Haibo Zhou, Lin Gui, and Xuemin Shen. Cooperative task scheduling for computation offloading in vehicular cloud. *IEEE Trans. Veh. Technol.*, 67(11):11049–11061, 2018.

[48] Yuxuan Sun, Xueying Guo, Jinhui Song, Sheng Zhou, Zhiyuan Jiang, Xin Liu, and Zhisheng Niu. Adaptive learning-based task offloading for vehicular edge computing systems. *IEEE Trans. Veh. Technol.*, 68(4):3061–3074, 2019.

[49] Teppei Suzuki, Keisuke Ozawa, and Yusuke Sekikawa. Rethinking pointnet embedding for faster and compact model. In *Proc. IEEE 3DV'20*, pages 791–800, 2020.

[50] Gokulnath Thandavarayan, Miguel Sepulcre, and Javier Gozalvez. Analysis of message generation rules for collective perception in connected and automated driving. In *Proc. IEEE IV'19*, pages 134–139, 2019.

[51] Gokulnath Thandavarayan, Miguel Sepulcre, and Javier Gozalvez. Generation of cooperative perception messages for connected and automated vehicles. *IEEE Trans. Veh. Technol.*, 69:16336–16341, 2020.

[52] Shaohua Wan, Xiaolong Xu, Tian Wang, and Zonghua Gu. An intelligent video analysis method for abnormal event detection in intelligent transportation systems. *IEEE Trans. Intell. Transp. Syst.*, 22(7):4487–4495, 2020.

[53] Yicong Wang, Gustavo De Veciana, Takayuki Shimizu, and Hongsheng Lu. Performance and scaling of collaborative sensing and networking for automated driving applications. In *Proc. IEEE ICC'18*, pages 1–6, 2018.

[54] DoHyun Daniel Yoon, Beshah Ayalew, and GG Md Nawaz Ali. Performance of decentralized cooperative perception in v2v connected traffic. *IEEE Trans. Intell. Transp. Syst.*, 2021. to appear, doi: 10.1109/TITS.2021.3063107.

[55] Fuxin Zhang, Guozhen Tan, Chao Yu, Nan Ding, Caixia Song, and Mingjian Liu. Fair transmission rate adjustment in cooperative vehicle safety systems based on multi-agent model predictive control. *IEEE Trans. Veh. Technol.*, 66(7):6115–6129, 2016.

[56] Jun Zhang and Khaled B Letaief. Mobile edge intelligence and computing for the internet of vehicles. *Proc. IEEE*, 108(2):246–261, 2019.

[57] Shan Zhang, Jiayin Chen, Feng Lyu, Nan Cheng, Weisen Shi, and Xuemin Shen. Vehicular communication networks in the automated driving era. *IEEE Commun. Mag.*, 56(9):26–32, 2018.

[58] Xiao Zhang, Wenda Xu, Chiyu Dong, and John M Dolan. Efficient L-shape fitting for vehicle detection using laser scanners. In *Proc. IEEE IV'17*, pages 54–59, 2017.

[59] Xinran Zhang, Zhimin He, Yaohua Sun, Shuo Yuan, and Mugen Peng. Joint sensing, communication, and computation resource allocation for cooperative perception in fog-based vehicular networks. In *Proc. IEEE WCSP'2021*, pages 1–6, 2021.

[60] Xiangmo Zhao, Pengpeng Sun, Zhigang Xu, Haigen Min, and Hongkai Yu. Fusion of 3D lidar and camera data for object detection in autonomous vehicle applications. *IEEE Sens. J.*, 20(9):4901–4913, 2020.

[61] Zhongling Zhao, Jia Shi, Zan Li, Jiangbo Si, Pei Xiao, and Rahim Tafazolli. Multi-objective resource allocation for mmwave MEC offloading under competition of communication and computing tasks. *IEEE Internet Things J.*, 2021. to appear, doi: 10.1109/JIOT.2021.3116718.

[62] Han Zheng, Ruisheng Wang, and Sheng Xu. Recognizing street lighting poles from mobile LiDAR data. *IEEE Trans. Geosci. Remote. Sens.*, 55(1):407–420, 2016.

[63] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proc. IEEE CVPR'18*, pages 4490–4499, 2018.

[64] Ying Zhou, Dan Wang, Xiang Xie, Yiyi Ren, Guolin Li, Yangdong Deng, and Zhihua Wang. A fast and accurate segmentation method for ordered LiDAR point cloud of large-scale scenes. *IEEE Geosci. Remote. Sens. Lett.*, 11(11):1981–1985, 2014.

[65] Hao Zhu, Ka-Veng Yuen, Lyudmila Mihaylova, and Henry Leung. Overview of environment perception for intelligent vehicles. *IEEE Trans. Intell. Transp. Syst.*, 18(10):2584–2601, 2017.

Appendix

Appendix A

SOCP Problem Transformation

Introduce two auxiliary continuous decision variable sets, $\boldsymbol{\psi}^{\mathtt{A}} = \{\psi_n, \forall n \in \mathcal{N}^{\mathtt{A}}\}$ and $\boldsymbol{\theta}^{\mathtt{A}} = \{\theta_{n,n'}, \forall (n,n') \in \mathcal{L}^{\mathtt{A}}\}$, and one auxiliary continuous decision variable, ξ. Then, the nonlinear and non-convex resource allocation optimization problem $\mathbf{P}2$ can be transformed to an SOCP problem with zero optimality gap, given by

$$
\begin{aligned}
\mathbf{P}4: \quad \min_{\boldsymbol{\alpha}^{\mathtt{A}},\boldsymbol{\beta}^{\mathtt{A}},\boldsymbol{\psi}^{\mathtt{A}},\boldsymbol{\theta}^{\mathtt{A}}} \quad & \omega B \sum_{(n,n')\in\mathcal{L}^{\mathtt{A}}} \beta_{n,n'} + (1-\omega) \sum_{n\in\mathcal{N}^{\mathtt{A}}} \alpha_n f_n \\
\text{s.t.} \quad & \sum_{(n,n')\in\mathcal{L}^{\mathtt{A}}} \beta_{n,n'} \leq 1 \\
& C_{n,n'}\theta_{n,n'} + C_{n'}\psi_{n'} \leq T \quad \forall (n,n') \in \mathcal{L}^{\mathtt{A}} \\
& \beta_{n,n'}\theta_{n,n'} \geq \xi^2 \quad \forall (n,n') \in \mathcal{L}^{\mathtt{A}} \qquad \text{(A1)}
\end{aligned}
$$

$$\alpha_{n'}\psi_{n'} \geq \xi^2 \quad \forall n \in \mathcal{N}^{\mathtt{A}} \tag{A2}$$

$$\xi = 1$$

$$0 < \beta_{n,n'} \leq 1, \quad \forall (n,n') \in \mathcal{L}^{\mathtt{A}}$$

$$0 < \alpha_n \leq 1, \quad \forall n \in \mathcal{N}^{\mathtt{A}}$$

The fundamental difference between problems $\mathbf{P}2$ and $\mathbf{P}4$ lies in "$\geq$" sign in rotated second order cone constraints (A1) and (**??**). If both constraints are active in an SOCP optimum, the SOCP optimum is also an optimum of problem $\mathbf{P}2$. Assume that there is an inactive constraint (A1) in an SOCP optimum, i.e., $\beta^{\star}_{n,n'}\theta^{\star}_{n,n'} > 1$. If $\theta^{\star}_{n,n'}$ is replaced by $\theta^{\circ}_{n,n'}$, with $\theta^{\circ}_{n,n'} < \theta^{\star}_{n,n'}$ and $\beta^{\star}_{n,n'}\theta^{\circ}_{n,n'} = 1$, all constraints are still satisfied, and the objective value is unchanged, inferring that $[\beta^{\star}_{n,n'}, \theta^{\circ}_{n,n'}]$ is an optimal pair in another SOCP optimum. Similar conclusions can be made for constraint (**??**). Therefore, for a feasible SOCP problem $\mathbf{P}4$, there is always an optimum with active constraints (A1) and (**??**), which is also the optimum of problem $\mathbf{P}2$.