Online Vehicle Subgroup Scheduling for Feature-Level Cooperative Classification

by

Zhenhuan Sun

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 Computer Engineering

Waterloo, Ontario, Canada, 2023

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

In future transportation systems, autonomous vehicles (AVs) are expected to operate under complex driving environments without human intervention. Their ability to continuously perceive and understand the surrounding environment can be realized through the execution of a range of environment perception tasks, including object classification, object detection, and object tracking. In this thesis, we investigate an autonomous driving scenario wherein a group of AVs is tasked with tracking a common object over a period of time. To tackle issues related to the inefficient utilization of global sensing data and excessive consumption of computing resource in conventional standalone tracking, a cooperative classification scheme has been proposed as a replacement for the conventional non-cooperative classification scheme employed in the tracking process. Taking into account the dynamic nature of operating environments and the varying availability of computing resources, a subgroup scheduling problem is studied to optimize the performance of the cooperative classification scheme. Our objective is to determine the composition and role assignment of each scheduled subgroup in order to minimize the total computation demand required by the group for performing cooperative classifications, while ensuring the satisfaction of classification accuracy and delay requirements. A learning-based solution is developed, which integrates multi-armed bandit (MAB) theory with distance and line-of-sight based subgroup selection criteria, to guide subgroup scheduling decisions in the presence of randomness and uncertainties. Simulation results confirm the effectiveness of our proposed scheme and algorithm, outperforming other baseline scheme and algorithms by delivering improved classification accuracy and reduced classification delays with lower computation

demand.

Acknowledgements

I would like to express my utmost gratitude to Professor Weihua Zhuang, my supervisor, for providing me with the research opportunity and for her guidance and support throughout my master's studies. Professor Zhuang's advice during research discussions have been truly invaluable and thought-provoking, helping me hone my research skills and fostering my growth as an independent researcher. Her advice and guidance made this thesis possible.

I would like to extend my gratitude to Professor Xuemin Shen for organizing the weekly group meetings. His advice on presentation skills and research methodologies have been instrumental in improving my ability to effectively communicate research results and cultivating the essential qualities I need to become a qualified researcher. Furthermore, I want to thank both Professor Xuemin Shen and Professor Zhou Wang for serving as my thesis readers. Their valuable comments and suggestions have greatly enriched this thesis.

I am fortunate to have worked with Dr. Kaige Qu, whose constant support and insightful advice have greatly contributed to my research. I would like to express my sincere gratitude to her, without whom this thesis would not have been possible. I would also like to express my appreciation to my fellow colleagues at the BBCR lab. Their expertise and advice have greatly contributed to my growth as a researcher and have enriched my understanding of research.

Lastly, I am grateful for the immeasurable love bestowed upon me by my beloved parents, my mother Min Xia and my father Yufeng Sun. Their unwavering support and encouragement have been the guiding force behind my achievements, and for that, my

heart overflows with gratitude.

Dedication

This thesis is dedicated to my parents, Yufeng Sun and Min Xia.

Table of Contents

Author's Declaration ii

Abstract iii

Acknowledgements v

Dedication vii

List of Figures xi

List of Tables xiii

List of Abbreviations xiv

List of Symbols xv

1 Introduction 1

1.1 Issues with Stand-alone Intelligence 4

1.2 Existing Solutions and Motivation 5
1.3 Proposed Solution 7
1.4 Challenges and Objective 11
1.5 Organization of the Thesis 16

2 Literature Survey **17**

2.1 Cooperative Perception 17
2.2 Multi-view Classification Networks 19
2.3 Classification Performance Estimation 21
2.4 Global State Information Estimation 22
2.5 Summary 23

3 System Model **24**

3.1 Autonomous Driving Scenario 24
3.2 Task Model 26
3.3 Computing Model 29
3.4 Communication Model 31
3.5 Cooperative Classification Accuracy 33
3.6 Summary 34

4 Problem Formulation and Solution **35**

4.1 Problem Formulation 35

4.2 Problem Solution 40

4.2.1 Overview 40

4.2.2 Viewing Condition Assessment 43

4.2.3 Identification of Superior Subgroup 49

4.2.4 Subgroup Exploration Space Reduction 52

4.2.5 Explore-then-Commit 54

4.3 Summary 60

5 Performance Evaluation **61**

5.1 Simulation Setup 61

5.2 Simulation Results 67

5.3 Summary 78

6 Conclusion and Future Work **80**

6.1 Conclusion 80

6.2 Future Research Work 81

References **84**

List of Figures

1.1 An illustration of the considered autonomous driving scenario and its associated traffic participants. 2

1.2 An illustration demonstrating the periodic updates of object information over the tracking period. Object classification is performed on each update frame, with the requirement that the classification result must be obtained within the maximum tolerable delay. Object tracking is conducted on every frame that falls between consecutive update frames. 3

1.3 An illustration of role assignment in the group of AVs for cooperative classification. 7

1.4 An illustration of cooperative classification scheme. 8

1.5 An illustration of role assignment in the group of AVs for improved cooperative classification scheme. 9

3.1 An illustration of time slots for classifications. 25

4.1 An illustration of parameters within the occlusion estimation algorithm. . . 46

5.1 An example of the spatial arrangement of a group of 8 AVs (blue) and the object of interest (green) within the synthetic driving scenario. 62

5.2 An example of how the images of the object of interest are processed for AVs under different viewing conditions. AVs with clear, unobstructed line of sight to the object of interest are depicted in blue, while AVs face obstructions in their line of sight are depicted in red. 64

5.3 Average classification performance per time slot for group of 4 AVs 69

5.4 Average classification performance per time slot for group of 8 AVs 71

5.5 Comparison of average classification accuracy across strategies and group sizes in two simulation settings 73

5.6 Comparison of average classification delay across strategies and group sizes in two simulation settings 75

5.7 Comparison of total computation demand across strategies and group sizes in two simulation settings 77

List of Tables

5.1 Simulation parameters. 63

5.2 Feature Extraction and Classification Modules 65

List of Abbreviations

AVs	autonomous vehicles
CNNs	convolutional neural networks
CP	cooperative perception
DNNs	deep neural networks
ETSI	European Telecommunications Standards Institute
GVCNN	group-view convolutional neural network
LCB	lower confidence bound
MAB	multi-armed bandit
mmWave	millimeter wave
MVCNN	multi-view convolutional neural network
QoS	quality of service
UCB	upper confidence bound
V2I	vehicle-to-infrastructure
V2V	vehicle-to-vehicle
V2X	vehicle-to-everything

List of Symbols

a	An instance of subgroup tuple from subgroup set $\mathcal{A}$
$a_r^*(t)$	The subgroup with the highest LCB on the mean reward estimation at time slot t
$\mathcal{A}^*(t)$	Estimation of feasible subgroup set $\mathcal{A}^*$ at time slot t
$\mathcal{A}^*$	Set of all feasible subgroups
A_0	Path-loss coefficient
a_t	The subgroup selected from subgroup set $\mathcal{A}$ at time slot t for cooperative classification
$\mathcal{A}$	Set of all possible candidate subgroups
B	Total bandwidth
$\mathcal{C}_n(\mathcal{V}^{(C)})$	Set that contains all combinations of vehicles in $\mathcal{V}^{(C)}$ with size n
$\mathcal{C}_{n_r}(\mathcal{V}^{(O)})$	Set that contains all combinations of vehicles in $\mathcal{V}^{(O)}$ with size n_r

c_{t,a_t}	The cost incurred from selecting subgroup a_t at time slot t
$d^C(t)$	Total computing delay experienced by subgroup $\mathcal{K}_s(t)$ for executing feature aggregation and classification modules at time slot t
$d^E(t)$	Total computing delay experienced by subgroup $\mathcal{K}_s(t)$ for executing feature extraction module at time slot t
$d_k^E(t)$	Computing delay for executing feature extraction module at AV k at time slot t
$d_{\mathcal{K}_s(t)}$	Sum delay experienced by subgroup $\mathcal{K}_s(t)$ at time slot t for performing cooperative classification
$d^T(t)$	Total feature transmission delay incurred by helpers in subgroup $\mathcal{K}_s(t)$ at time slot t
$d_k^T(t)$	Transmission delay for transmitting feature maps of helper k to the aggregator $k_A(t)$ at time slot t
D_{far}	Far distance threshold from which the object classification performance starts declining
D_{near}	Near distance threshold from which the object classification performance starts declining
d	Distance between helper AV and aggregator AV
$f_k(t)$	Available CPU frequency AV k has for the cooperative classification at time slot t
F_k	Maximum CPU frequency of AV k

$\mathcal{G}_n$	Set that contains subgroups of size n with superior viewing conditions
$g_k(t)$	Transmission power gain between helper AV k and the aggregator $k_A(t)$ at time slot t
h_l	Height and width of filter in `CONV` layer l
H_l^I, W_l^I, D_l^I	Height, width, and depth dimensions of input feature maps of `CONV` layer l
H_l^O, W_l^O, D_l^O	Height, width, and depth dimensions of output feature maps of `CONV` layer l
$k_A(t)$	Aggregator AV in subgroup $\mathcal{K}_s(t)$ at time slot t
$\mathcal{K}_s^{(h)}(t)$	Set of helper AVs in subgroup $\mathcal{K}_s(t)$ at time slot t
K	Number of AVs in group $\mathcal{K}$
$\mathcal{K}$	Group of AVs
k	Index of AVs in group $\mathcal{K}$
$\mathcal{K}_s(t)$	Selected subgroup of AVs at time slot t
$\hat{l}$	Last layer of the feature extraction module
$\mathcal{L}^A$	Layer set of feature aggregation module
$\mathcal{L}^C$	Layer set of classification module
$\mathcal{L}^E$	Layer set of feature extraction module
l	Layer index of multi-view classification network
$n_a(t)$	The total number of times subgroup a has been selected until time slot t
p	Transmission power

r_{t,a_t}	The reward received from selecting subgroup a_t at time slot t
$S_k^{(D)}$	Distance-based score for AV k
$S_{max}^{(D)}$	Maximum cumulative distance score
t	Index of each time slot
T	Total number of time slots
v_l	The number of output data elements at layer l
$\mathcal{V}^{(C)}$	Set that contains AVs with clear views, ranked based on their distance-based scores $S_k^{(D)}$
$\mathcal{V}^{(O)}$	Set that contains AVs with obstructed views, ranked based on their distance-based scores $S_k^{(D)}$
w, l	Width and length of vehicle
(x_B, y_B)	Bottom-left vertex of vehicle bounding box
X_l^I	Input dimensions of `FC` layer l
X_l^O	Output dimensions of `FC` layer l
(x_o, y_o)	Position of the object of interest
(x_s, y_s)	Position of source vehicle
(x_T, y_T)	Top-right vertex of vehicle bounding box
(x_k, y_k)	Position of AV k
τ	Maximum tolerable delay for classification
$\psi_k(t)$	Transmission data size of helper AV k at time slot t after compression
ψ	Transmission data size before compression

$\phi_{\mathcal{K}_s(t)}$	Cooperative classification accuracy obtained by a subgroup realization $\mathcal{K}_s(t)$ at time slot t
Θ	Minimum classification accuracy requirement
Υ	Reward value
ρ	Subsidy factor
$\hat{\mu}_{c,a}(t)$	Mean cost estimation of subgroup a at time slot t
$\mu_{c,a}^{LCB}(t)$	Estimation of the lower limit of mean cost for subgroup a at time slot t
$\hat{\mu}_{r,a}(t)$	Mean reward estimation of subgroup a at time slot t
$\mu_{r,a}^{LCB}(t)$	Estimation of the lower limit of mean reward for subgroup a at time slot t
$\mu_{r,a}^{UCB}(t)$	Estimation of the upper limit of mean reward for subgroup a at time slot t
σ^2	Received noise power
$\boldsymbol{\alpha}(t)$	Binary subgroup member indicator vector at time slot t
$\alpha_k(t)$	Binary indicator indicating whether AV k is a member of subgroup $\mathcal{K}_s(t)$ at time slot t
$\boldsymbol{\gamma}(t)$	Binary aggregator indicator vector at time slot t
$\gamma_k(t)$	Binary indicator indicating whether AV k is the aggregator of subgroup $\mathcal{K}_s(t)$ at time slot t
Γ_{far}	Decay rate of $S_k^{(D)}$ at far distance threshold
Γ_{near}	Decay rate of $S_k^{(D)}$ at near distance threshold

π_l	The number of floating-point operations for computing one output data element at layer l
ω	Computation intensity representing the number of CPU cycles required to perform one floating-point operation
λ_l	Computation workload of layer l
$\zeta_{\mathcal{K}_s(t)}$	Total computation effort required by a subgroup $\mathcal{K}_s(t)$ to execute cooperative classification task at time slot t
$\zeta_k(t)$	Computation effort that an AV k put into executing cooperative classification at time slot t
κ	Energy efficiency coefficient
δ	The number of bits required to represent a floating-point number

Chapter 1

Introduction

Autonomous vehicles (AVs), as the indispensable component in realizing safe and reliable autonomous driving, are expected to play a major role in shaping the future of transportation. The advancements in sensor technologies, computer vision algorithms, and computing paradigms have resulted in a significant improvement in the ability of AVs to handle complex driving situations. This progress is expected to lead to a shift towards higher levels of automation in the near future, with level 4 or level 5 automation becoming a reality [1]. At such high levels of automation, AVs are anticipated to have the ability to independently perform all essential driving tasks without the need for human intervention or assistance from other vehicles or infrastructure. To ensure an accurate understanding of the dynamic driving environment and maintain driving safety throughout the entire autonomous driving session, the capability to consistently and accurately perceive the environment in real-time is of utmost importance for AVs. Therefore, environment perception tasks such as object classification, detection, and tracking have been extensively studied in

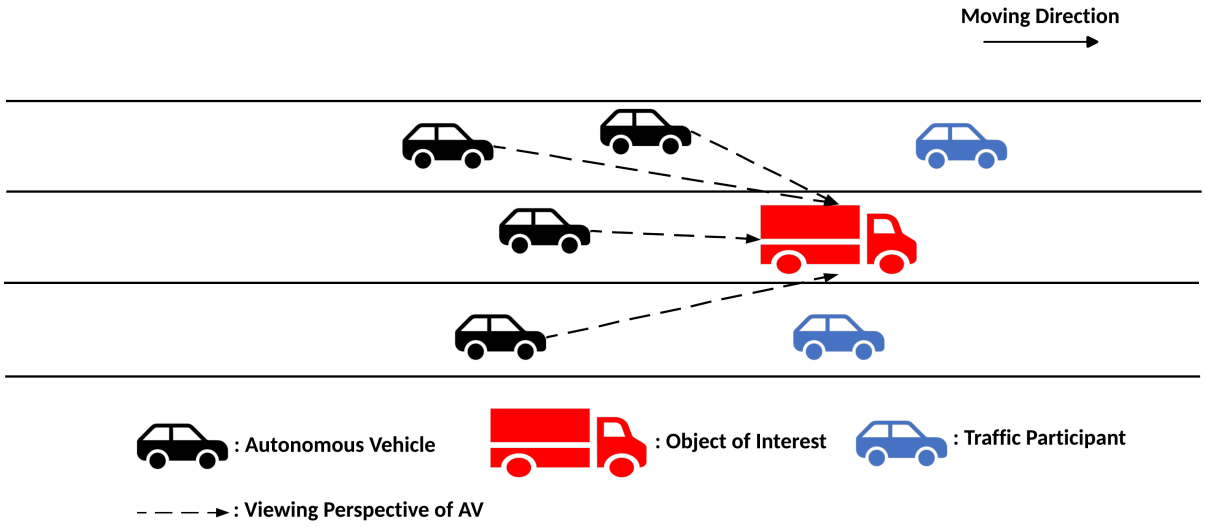


Figure 1.1: An illustration of the considered autonomous driving scenario and its associated traffic participants.

the context of autonomous driving to provide AVs the capability to operate safely and reliably in a range of driving scenarios [2–5]. Consider an autonomous driving scenario where a group of closely positioned AVs, coordinated by a group head, must track an object for a certain period of time. As depicted in Figure 1.1, multiple AVs in proximity, each perceiving a different view of a moving truck, need to follow the truck for a certain period due to their shared temporary destination. Because of the potential danger the truck and its cargo pose to both the AVs and other traffic participants on the road, all AVs must continuously track the truck, monitor its behavior, and take appropriate actions, such as keeping a safe distance from the truck or applying brakes in the event of a sudden stop, throughout the period of following. During the process of tracking, as each AV captures real-time video frames of the object, the appearance of the tracked object undergoes continuous changes caused by factors such as occlusion or variation in illumination. To enable effective tracking

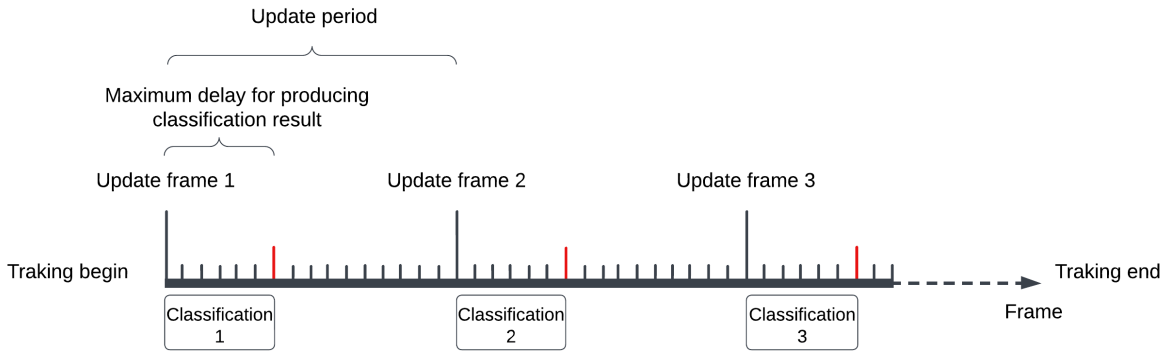


Figure 1.2: An illustration demonstrating the periodic updates of object information over the tracking period. Object classification is performed on each update frame, with the requirement that the classification result must be obtained within the maximum tolerable delay. Object tracking is conducted on every frame that falls between consecutive update frames.

in a dynamic environment, simultaneous utilization of object detector and object tracker is essential. The object tracker operates on each frame, computing optical flow between extracted feature points or similarity in motion or appearance of the object across different frames to track the detected object [6], [7]. However, the tracking accuracy gradually diminishes over time due to accumulated tracking errors caused by occlusion or changes in the appearance of the target object. As a result, detections are performed periodically on update frames to update the information regarding the object and compensate for tracking errors caused by the time-varying nature of the environment [6, 8, 9]. To facilitate safe and reliable execution of real-time tracking, the object detector must locate and classify the object in update frames with sufficient accuracy and within an acceptable delay to ensure that the updated information regarding the object can be promptly available for tracking in the subsequent frames [10], [11]. In this work, we focus on the object classification

aspect of the tracking process, and investigate a process where each AV needs to deliver a series of accurate and prompt classification of the object throughout the tracking period. Such process is depicted in Figure 1.2.

1.1 Issues with Stand-alone Intelligence

A conventional scheme to execute classifications involves each AV operating independently. In this scheme, each AV individually processes its own sensing data and performs local computation related to each object classification task. This decentralized scheme allows each AV to make decisions autonomously without relying on central coordination or information sharing with other vehicles. However, if AVs rely solely on their local sensing data acquired from a single viewpoint, their classification accuracy will be susceptible to viewing conditions. For example, an AV with limited viewing angle may fail to capture all the necessary information about the object, particularly if its view is partially or fully occluded, or it is located in a poorly-lit environment. In addition, in this scheme, each AV operates independently and must undertake a multitude of computation-intensive tasks associated with object classification. These tasks include complex processes such as multi-modal sensing data fusion and feature extraction, which are crucial for accurate and reliable classification of objects and necessitate significant computing resources for effective execution [12], [13]. Therefore, the overall computation demand for the group of AVs can be substantial and the total computation workload of the group scales proportionally with the number of AVs, leading to considerable consumption of computing resources over the tracking period [14].

1.2 Existing Solutions and Motivation

To improve classification accuracy and reliability over time, the utilization of multi-view sensing data can be advantageous [15], [16]. By providing a set of diverse views of the object, multi-view sensing data can complement each other and reveal important features and attributes that might not be discernible from a single viewpoint. For example, if a truck is viewed from both side and back, its shape and type may be more easily discernible than if viewed only from the back. As a result, by leveraging the vehicle-to-everything (V2X) communication network to facilitate the sharing and aggregation of sensing data from different AV, cooperative perception (CP) techniques have emerged as a promising paradigm for perception in the autonomous driving context [15, 17–22]. In the existing literature, three types of CP have been identified, namely object-level, feature-level, and raw-level CP. Each type of CP leverages different forms of multi-view sensing data and comes with its own set of advantages and limitations. To achieve notable improvements in perception accuracy and reliability, it is preferable to adopt raw-level or feature-level CP techniques, which entails the sharing and aggregation of unprocessed raw sensing data or extracted features from the raw sensing data [18], [20]. However, such techniques typically impose a burden on the environment with limited communication resources due to the considerable data size of both raw sensing data or feature data [21], [22]. Transmitting the data often requires more bandwidth or lead to prolonged transmission times, making it inherently communication inefficient and challenging to implement in resource-constrained settings.

Excessive computing resource consumption issue can be mitigated by leveraging com-

putation reuse. By sharing reusable computation results (e.g., classification result of the object of interest) from one device, other devices which require the same computation result can reuse the computed result without undertaking additional computations. This prevents the execution of duplicate computations and effectively reduces the overall computing demand and computing resource consumption [23], [24]. However, such method lacks the utilization of multi-view sensing data and relies input data from a single device to produce reusable computation results. Employing computation reuse in the considered scenario would once again result in unreliable classification accuracy over time.

In summary, to tackle the issues concerning vulnerability to viewing conditions and high computing resource consumption in stand-alone intelligence, CP techniques have facilitated the utilization of multi-view sensing data to enhance classification accuracy and reliability. However, the lack of communication efficiency limits their full potential and effectiveness in practical implementations. While the utilization of computation reuse can lead to a reduction in computing demand and computing resource consumption, it does not fully leverage the advantages offered by multi-view sensing data. Therefore, motivated by the issues of existing scheme and the limitations of existing methods, we aim to explore a scheme that allows AVs to efficiently leverage multi-view sensing data in each classification of the tracking period, while reducing both the computing demand and the communication overhead incurred in each classification.

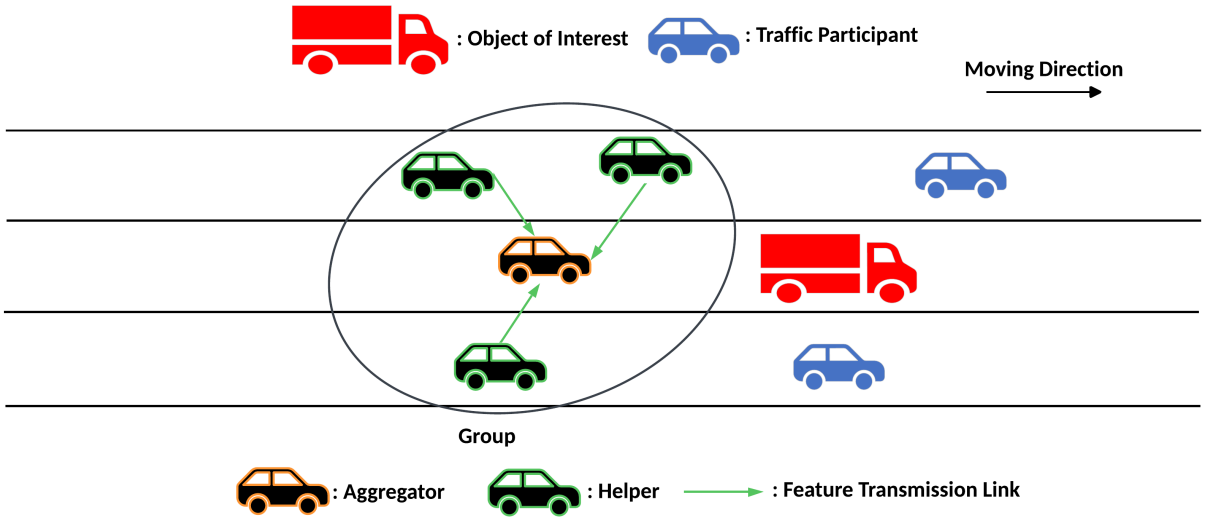


Figure 1.3: An illustration of role assignment in the group of AVs for cooperative classification.

1.3 Proposed Solution

Multi-view classification methods that employ image-based deep convolutional neural networks (CNNs), termed as multi-view convolutional neural network (MVCNN), have attracted our attention for its ability to effectively leverage multi-view sensing data and reduce computing demand in the classification process. Extensive research efforts have been devoted to exploring and advancing these methods over the past decade, driven by its outstanding performance in tasks such as object classification and retrieval [25–32]. By resorting to vehicle-to-vehicle (V2V) communications and multi-view classification methods, view images acquired from different AVs can be effectively utilized. Simultaneously, the three modules of multi-view classification methods, including the feature extraction module, feature aggregation module, and classification module, can be distributed across

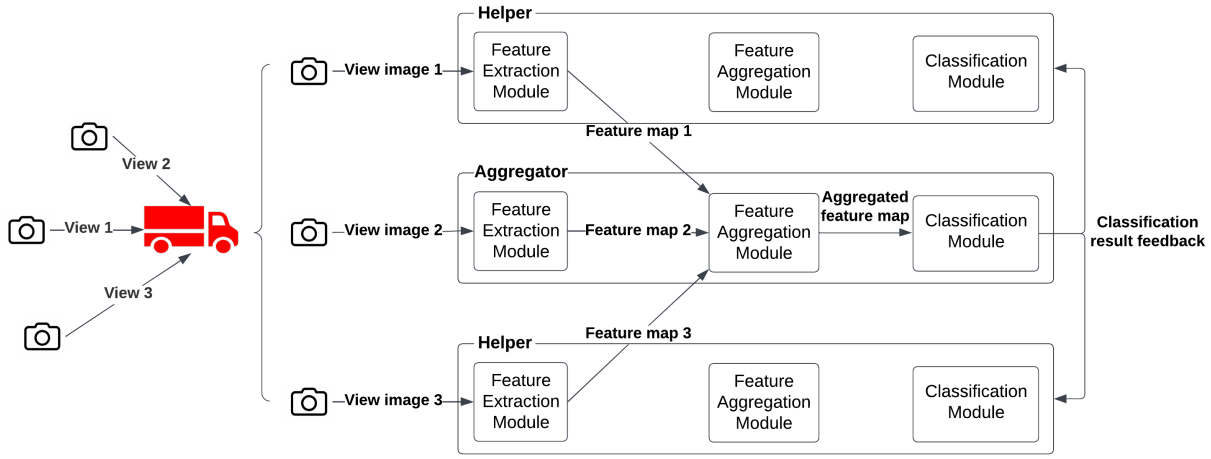


Figure 1.4: An illustration of cooperative classification scheme.

multiple vehicles, forming a distributed computing hierarchy that allows vehicles to cooperatively generate classification outcomes [33], [34]. As a result, a cooperative classification scheme can be realized for classifications during the tracking period. In such scheme, each AV will be assigned a role, either as a *helper* or an *aggregator*. The distribution of roles for AVs in the group is illustrated in Figure 1.3. The aggregator is responsible for executing all three modules, whereas the helpers are solely responsible for executing the feature extraction module. For each cooperative classification of the object over the tracking period, as depicted in Figure 1.4, both helpers and aggregator utilize the feature extraction module of MVCNN to extract view-based features from their locally captured image of the object. The helpers then transmit their extracted features to the aggregator through V2V communication. Upon receiving the features from helpers, the aggregator aggregates the received features with its own extracted features using the feature aggregation module and forwards the resulting aggregated features to the classification module for final classi-

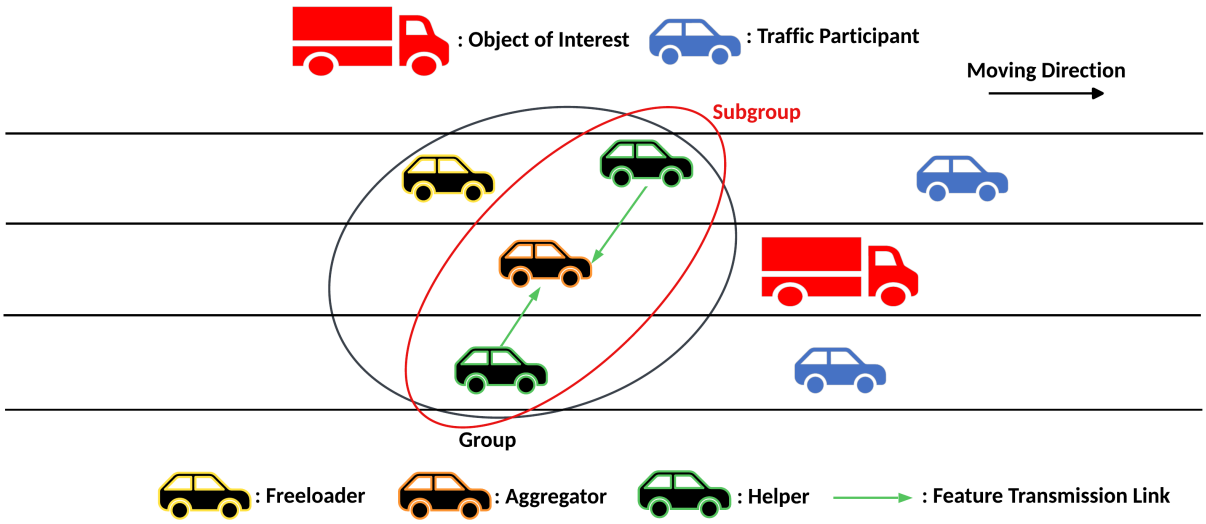


Figure 1.5: An illustration of role assignment in the group of AVs for improved cooperative classification scheme.

fication. After obtaining the classification result, the aggregator shares the result with all AVs in the group. This cooperative classification process is repeated for each classification of the object throughout the tracking period. As a result, instead of performing all the computation associated with each classification (e.g., feature extraction module and classification module), most devices (e.g., helpers) only perform a portion of them (e.g., feature extraction module). By adopting this cooperative classification scheme, computations and computing resources required by multiple AVs to classify the same object can be significantly reduced, while the classification accuracy and reliability are improved through the utilization of multi-view sensing data.

Further improvements can be made to this scheme by reducing the number of AVs involved and allowing only a selected subgroup of AVs to perform cooperative classification. This intuition stems from the realization that AVs' views may exhibit similarities

in certain circumstances [31]. For example, AVs located in the same lane may capture image of the object from similar angles, providing redundant information during cooperative classification. In contrast, views from AVs located in different lanes may provide more discriminative information about the object, thus contributing more in cooperative classification. Selecting a subgroup of AVs that offer informative and non-redundant views of the object may be sufficient to attain the same level of classification accuracy as when all AVs in the group are selected for cooperative classification. In the improved cooperative classification scheme, a subgroup of AVs is selected from the group to perform cooperative classification, while the remaining AVs assume the role of freeloaders [35]. The distribution of roles for the group of AVs in this improved scheme is illustrated in Figure 1.5. The subgroup follows the same workflow as the previous scheme for cooperative classification, with the key difference being that the aggregator of the subgroup also shares the cooperative classification results with freeloaders. Therefore, freeloaders do not need to actively participate in the cooperative classification process, but can still have access to the classification results obtained by the subgroup. The decrease in the number of AVs required to perform feature extraction and transmission in each cooperative classification further reduces the overall computation workload and alleviates communication burden. Additionally, the classification accuracy and reliability can remain unaffected as long as the AVs in the subgroup provide informative and non-redundant views of the object.

1.4 Challenges and Objective

In this work, we aim to schedule a subgroup for each cooperative classification over the tracking period. The objective is to determine the size and composition of each subgroup, and assign appropriate role (e.g., aggregator/helper) to each member, in order to meet the accuracy and delay requirements of each classification while minimizing the total computation demand required by the group for performing cooperative classifications. We have observed that choosing subgroups to achieve our objective is essentially a matter of managing the interplay among design parameters, including subgroup size, subgroup members, and role assignments of members, to balance the trade-off among cooperative classification accuracy, delay, and computation demand. One crucial design parameter in balancing such trade-off is the subgroup size. For example, expanding the subgroup size could potentially lead to more comprehensive and diverse observations of the object, which improve the accuracy of cooperative classification [30], [32]. However, the number of helpers in the subgroup increases with the subgroup size, and every additional helper will be required to execute feature extraction module and transmit extracted feature to the aggregator, which increases the total feature transmission delay and computation demand. A choice should be made between selecting a larger number of AVs to attain more accurate and reliable cooperative classification at the expense of increased computation demand and classification delay, and selecting a smaller number of AVs for faster and more computing-efficient classification with lower accuracy and reliability. On the other hand, the interplay among design parameters must also be considered when forming the subgroup. For example, through the careful selection of subgroup members with informative

and non-redundant views for cooperative classification, it is possible to achieve an equivalent level of cooperative classification accuracy by a smaller subgroup as compared with a larger subgroup consisting of AVs with inferior object views. This leads to a reduction in subgroup size and enables faster and more computing-efficient cooperative classification.

Managing the interplay among design parameters and balancing the trade-off among accuracy, delay, and computation demand requires the knowledge of cooperative classification accuracy and delay of each possible subgroup, and their corresponding computation workload and computing resource consumption for performing cooperative classification. However, acquiring this information beforehand can be challenging. On one hand, cooperative classification accuracy is challenging to estimate due to two primary reasons.

- **Inherent randomness in deep neural network**. The outstanding performance of multi-view classification methods can be attributed to the ability of deep neural networks (DNNs) to learn complex and representative features from input data [36]. However, due to the inherent complexity of DNNs, understanding and interpreting its feature learning process and decision-making process remain difficult. Thus, it is unclear what features are detected from input data and how features interact to produce classification result [37–40]. The ambiguity in the relationship between the input data, such as an image taken from a specific view, and the extracted features introduces randomness into the classification accuracy, making it difficult to determine the exact classification accuracy of an object. This randomness is exacerbated when images from multiple views are taken into account and a feature aggregation mechanism is considered. Thus, it is challenging to estimate the exact

cooperative classification accuracy of a subgroup.

- **Unpredictability in a dynamic driving environment**. In a driving environment, the presence of real-time changing factors, such as lighting conditions and traffic participants, causes the content of the captured image data to constantly change. For example, the intensity values of pixels in two images captured from the same view may differ due to changes in lighting. The changes in the collected data affect the classification accuracy of an object overtime in an unknown manner due to the unpredictability of environment conditions. This introduces another source of randomness in cooperative classification accuracy in addition to the inherent randomness in deep neural networks.

On the other hand, there are challenges associated with estimating cooperative classification delay of candidate subgroups.

- **Varying resource availability**. The time required for subgroup members to extract features, and for the aggregator to perform feature aggregation and classification, is contingent upon the CPU frequency available to the helpers and the aggregator at each cooperative classification. However, each AV may be concurrently running different computation tasks alongside the cooperative classification task, making it challenging to determine the exact CPU frequency that each AV can provide for the cooperative classification [41]. Therefore, estimating the computing delay of a subgroup for cooperative classification can be challenging.

- **Unpredictable compression ratio**. The transmission of uncompressed feature data imposes a heavy communication burden, a common practice is to compress the

extracted features before transmission [42], [43]. As a result, helpers of the subgroup may have varying transmitting data sizes, as the size of the compressed features may differ based on the data contained in the uncompressed features; for example, data with repetitive values may compress better compared to those with random values. Therefore, the transmission delay for cooperative classification can not be accurately predicted due to the lack of accurate information feature data compression ratio.

Existing literature has various methods to tackle these challenges. Regression techniques and lookup tables are employed in some studies to estimate the classification performance of DNN-based methods and mitigate the impact of uncertainty in DNN models on classification performance [10], [15]. To address the problem of lacking global state information on the available computing resource, channel conditions, and transmitting data size, beaconing messages have been employed to estimate the state of vehicles [44]. Section 2.3 and 2.4 provide a more detailed discussion on the existing methods employed to address these challenges, along with their limitations. Building upon this understanding, to accommodate the dynamic nature of driving environment and devise algorithm that is capable of making subgroup scheduling decisions under uncertainties, we resort to an online learning-based method. To be specific, we utilize the multi-armed bandit (MAB) theory to select appropriate subgroups for each cooperative classification over the tracking period [45]. The MAB theory seeks to strike a balance between exploration and exploitation of candidate actions in order to maximize the cumulative reward over a series of actions. Exploration entails experimenting with currently sub-optimal actions to gather more data about their reward distributions, while exploitation involves identifying the currently optimal action based on empirical evidence. The goal is to learn about the reward distribution associated

with each action by observing past action-reward outcomes, and utilize that knowledge to make the most beneficial decisions for future actions. The upper confidence bound (UCB) based algorithms have been extensively employed as effective solutions to balance the exploration and exploitation trade-off [45] [46]. These algorithms provide a robust performance guarantee and have been applied in wireless networks to learn the dynamics of unknown environments [41], [47]. In this work, we leverage MAB theory in conjunction with our insights on the selection of subgroup to schedule subgroups for cooperative classification. The contributions of this research are summarized as follows:

- A computation efficient cooperative classification scheme is proposed to address the issues concerning the susceptibility of classification accuracy to viewing conditions of vehicles and inefficient utilization of computing resources in the object tracking process of autonomous driving. The scheme involves the sharing of multi-view sensing data and the distribution of classification-related computation workload among AVs within a selected subgroup, which has the capability to enhance the computing efficiency and accuracy of each classification.

- A subgroup scheduling optimization problem is formulated to maximize the performance of the cooperative classification scheme, while taking the dynamic nature of the vehicle operating environment into consideration. We propose a learning-based solution, based on the cost-subsidized MAB theory and our developed insights on subgroup selection, to minimize the total computation demand required by the group for cooperative classifications while facilitating the satisfaction of classification accuracy and delay requirements.

- Extensive simulations are conducted under a synthetic scenario to evaluate the effectiveness of the proposed scheme and algorithm. Simulation results demonstrate the capability of the proposed scheme and algorithm to enable AVs complete classifications with low computation demand, while consistently achieving improved classification accuracy and reduced classification delay across driving scenarios with varying vehicle densities.

1.5 Organization of the Thesis

The remainder of this thesis is organized as follows. In Chapter 2, a literature review is conducted on cooperative perception techniques, multi-view classification networks, and existing solutions proposed to address the aforementioned challenges. The system model is described in Chapter 3. A subgroup scheduling optimization problem is formulated in Chapter 4, in which detailed explanations and discussions on the proposed algorithm are also provided. In Chapter 5, simulation results are presented to evaluate the efficacy of the proposed scheme and algorithm. Finally, Chapter 6 concludes this study and outlines future research directions.

Chapter 2

Literature Survey

2.1 Cooperative Perception

Cooperative perception plays a vital role in advancing the capabilities of autonomous driving systems by enabling vehicles to collectively perceive and understand their surrounding environment. Unlike traditional perception systems that rely solely on the sensors within an individual vehicle, cooperative perception leverages the power of vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communication to share and aggregate sensing data among multiple vehicles. The extended perception horizon and the diverse observation perspectives offered by CP make AVs' views less vulnerable to sensor impairments and occlusions, which enhance the accuracy, reliability, and efficiency of perception tasks such as object detection, tracking, and scene understanding [48], [49], [50].

CP encompasses three distinct types that leverage different forms of sensing data. Raw-

level CP involves the sharing and aggregation of unprocessed raw sensing data, allowing for the preservation and utilization of complete sensing information, which contributes to enhanced perception performance. However, the transmission of raw sensing data presents a significant challenge in resource-constrained environments, as the large data size poses a burden on AVs in terms of real-time transmission capabilities [18], [22]. For example, the high data rate of millimeter wave (mmWave) communications is exploited in [51] and [52] to support the transmission of massive raw sensing data. On the other hand, object-level CP focuses on aggregating object detection results from multiple AVs to extend perception horizon. This approach offers simplicity and communication efficiency since it only requires the transmission of detection results. However, it is important to note that object-level CP cannot surpass the collective sensing capability of all participating AVs. In scenarios where none of the AVs can detect an object using their own sensing data, object-level CP alone will not be able to detect the object, as it relies solely on the sharing of detection results rather than the raw sensing data [16], [53]. Finally, feature-level CP leverages feature data extracted from the raw sensing data for aggregation. These feature data are lightweight compared to raw sensing data. For example, in the study conducted by Chen et al. [20], the data required for feature aggregation is only one hundredth of the size of the original data, showing that feature data is more communication-efficient than raw sensing data and better suited for real-time transmission. Despite their reduced data size, feature-level CP techniques have demonstrated a comparable level of enhancement in perception performance to raw-level CP techniques in certain applications, effectively striking a balance between accuracy and communication bandwidth requirements [20], [54]. Overall, understanding the different types of CP and their trade-offs is crucial for designing

cooperative perception systems that meet the specific requirements of resource-constrained environments, while achieving reliable and accurate perception capabilities.

2.2 Multi-view Classification Networks

Multi-view classification networks based on the deep learning techniques have gained significant attention in recent years due to their ability to leverage diverse visual perspectives for improved classification performance. In contrast to conventional single-view classification networks that rely on a single input image, multi-view classification networks leverage multiple views or images of the same object to substantially improve the accuracy and robustness of 3D object classifications [55], [56]. The underlying idea behind multi-view classification networks is to exploit the complementary information provided by different views to capture a more comprehensive representation of the object and achieve more reliable classification results. Inspired by the most pioneering work on multi-view object classification: MVCNN [32], most existing multi-view classification networks adopt a similar three-module architecture to generate classification results. Specifically, multi-view images of a 3D object are fed separately into the feature extraction module to extract view-specific features of the object for each view. The feature aggregation module collects and apply view-pooling operation to combine the extracted features from all views, resulting in the generation of multi-view aggregated feature representation of the 3D object. The aggregated features are then passed through the classification module to produce the final classification result. Different view-pooling operation can be applied for feature aggregation in multi-view classification. Inspired by the maxout operation in CNN [57],

MVCNN adopts element-wise maximum operation across the views to emphasize most salient features from all views and remove trivial ones [20]. Element-wise mean operation, which takes the average of each component of the feature maps, is often considered as an alternative for feature aggregation. However, it has been demonstrated to have inferior performance compared to max-pooling [32]. Arnold et al. [58] proposed the use of concatenation operation to merge all feature maps into a single feature map, retaining all the information for subsequent classification. While this operation has demonstrated superior performance compared to the other two operations, it expands the dimensionality of the resulting aggregated feature maps and necessitates modifications in the classification module, such as the addition of an extra linear layer to map the aggregated features back to the original input dimension, to ensure its proper functioning [33]. Over the years, MVCNN has undergone refinements and enhancements to further improve its performance and broaden its applicability [26, 27, 30, 31]. For example, Feng et al. [31] have considered the similarity between views and propose the group-view convolutional neural network (GVCNN) based on MVCNN to exploit the correlation and discriminability among different views. Yang et al. [30] observed that the view-pooling operation in MVCNN disregards the spatial correlation between 2D appearances captured from different viewpoints. To address this limitation, they proposed a Relation Network that explores both the region-to-region and view-to-view relationships, which leads to improved classification accuracy and enhanced performance consistency. Wang et al. [59] introduced a novel approach of employing dominant set clustering to group similar views together and pool information from different clusters. This approach allows for the utilization of informative features from disparate views, thereby enhancing the overall performance of the multi-view classification system.

2.3 Classification Performance Estimation

In order to assess the classification performance of DNN-based methods, researchers have employed various techniques. One common approach is the use of regression techniques, such as linear regression or quadratic regression, to determine a line of best fit that characterizes the relationship between classification performance metrics, e.g., accuracy or error probability, and factors that influence the performance, e.g., computation workload or input data size. In [15], Jia et al. adopt an empirical approach by assuming a logarithmic relationship between computation workload and detection performance. They fit the parameters using an image dataset and a pre-trained object detection model to obtain the best fit for this relationship. The established relationship is then utilized to estimate the detection performance based on the given computation workload. Zhao et al. [60] have experimentally shown that the inference accuracy of GoogleNet deep learning model grows with the pruned JPEG image size and the relationship can be fitted by a Gompertz function for future estimation. While regression techniques employed in these works are able to provide a line of best fit to estimate classification performance, they often focus on a single influencing factor. The complexity involved in accounting for all influencing factors may hinder their ability to accurately capture and model the impact of multiple factors, such as lighting conditions and viewing angles, on classification performance. Furthermore, the relationships captured by regression techniques are often static and may not adapt well to dynamic environments or changing conditions [61].

Alternatively, some researchers have opted to construct lookup tables, which establish a simple mapping between the classification performance and the influencing factors. This

approach provides a direct reference to determine the expected performance based on the given influencing factor. For example, in the studies conducted by Yang et al. [10], [62], the authors categorize the quality of captured video frames to "good" or "bad" and establish a mapping table to define the relationship between the inference error rate of a DL model and the input data quality. This mapping table enables the estimation of the performance of the model based on the quality of the input data. The advantage of lookup tables is their straightforward interpretation and ease of implementation. However, they may lack the flexibility to capture complex and nuanced relationships between the factors and performance, limiting their adaptability in dynamic environments.

2.4 Global State Information Estimation

In order to address the challenge of lacking global state information regarding available computing resources, channel conditions, and transmitting data size, different approaches have been adopted. Some works assume perfect knowledge of this information [63], [64], assuming that the devices can have access to accurate and up-to-date information. Other approaches rely on the periodic exchange of relevant information, such as resource availability and relative distance, among AVs to estimate the global state information [44], [65]. However, the frequent exchange of such information requires continuous communication among the AVs, which incurs high signaling overhead. Moreover, the dynamic nature of driving environment cause the road conditions and traffic participants to rapidly change over time [41]. Consequently, the exchanged information may quickly become stale and fail to reflect the current state of the environment. This stale information can undermine

the reliability and accuracy of the estimations based on it, as it may no longer align with the actual conditions [41], [66], [67].

2.5 Summary

In this chapter, we first introduce the benefits of utilizing cooperative perception techniques in modern autonomous driving systems, and a review is conducted on the three types of cooperative perception and their respective strengths and limitations. The three-module architecture of the multi-view classification networks and the corresponding operations performed by each module are described. A detailed comparison of different view-pooling operations, along with a brief review of recent advancements in the field of multi-view classification networks, are provided. We also examine the existing literature to explore how researchers have tackled the challenges described in Section 1.4, followed by a critical analysis of their proposed methods.

Chapter 3

System Model

3.1 Autonomous Driving Scenario

Consider an unidirectional multi-lane road where a group of K closely positioned AVs need to perform a series of classification on a common object of interest for a certain period of time. Let $\mathcal{K} = \{1, 2, \ldots, K\}$ denote the group of AVs, with $k \in \mathcal{K}$ representing the AV index. All AVs in group $\mathcal{K}$ are equipped with standardized built-in cameras that have uniform hardware specifications, enabling them to capture images of identical resolution and color depth for effective environmental perception. We consider a time slotted system, as depicted in Figure 3.1, comprising T time slots which represent a sequence of T classifications to be executed by AVs over the tracking period. Each time slot, indexed by t, has a duration of τ equal to the delay threshold of classification. We assume the composition of AVs within group $\mathcal{K}$ remains fixed throughout all time slots. At time slot

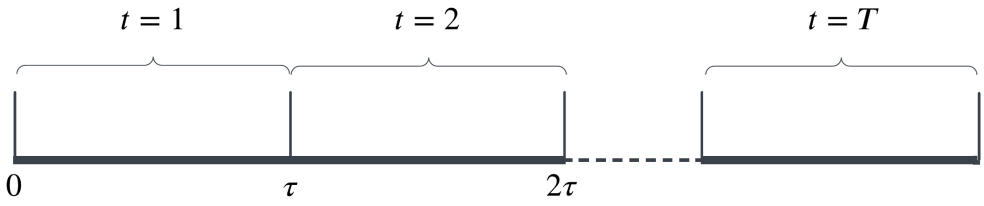


Figure 3.1: An illustration of time slots for classifications.

t, a subgroup of AVs is selected from group $\mathcal{K}$ to classify the object using cooperative classification scheme. Let $\boldsymbol{\alpha}(t) = \{\alpha_k(t), \forall k \in \mathcal{K}\}$ denote a binary subgroup member indicator vector in $\{0,1\}^K$, where $\alpha_k(t) = 1$ indicates that the AV k is selected as a member of the subgroup at time slot t, and $\alpha_k(t) = 0$ otherwise. We use $\mathcal{K}_s(t) = \{k \in \mathcal{K} \mid \alpha_k(t) = 1\}$ to denote the subgroup of AVs which are selected from set $\mathcal{K}$ at time slot t for cooperative classification. Within subgroup $\mathcal{K}_s(t)$, aggregator AV and helper AVs are identified. Let $\boldsymbol{\gamma}(t) = \{\gamma_k(t), \forall k \in \mathcal{K}\}$ denote a binary aggregator indicator vector in $\{0,1\}^K$, where $\gamma_k(t) = 1$ indicates that AV k serves as aggregator at time slot t, and $\gamma_k(t) = 0$ indicates that AV k serves as helper at time slot t. Let $\mathcal{K}_s^{(h)}(t) = \{k \in \mathcal{K} \mid \gamma_k(t) = 0\}$ denote the set of AVs that serve as helpers, and let $k_A(t)$ denotes the AV that serves as the aggregator. The relationship among sets $\{k_A(t)\}$, $\mathcal{K}_s^{(h)}(t)$, $\mathcal{K}_s(t)$, and $\mathcal{K}$ at any time slot t can be described as

$$\{k_A(t)\} \cup \mathcal{K}_s^{(h)}(t) = \mathcal{K}_s(t) \subseteq \mathcal{K}. \tag{3.1}$$

3.2 Task Model

The three-module architecture of multi-view classification framework typically consists of a view-pooling (`ViewPool`) layer and layers from an established image-based CNN model such as VGG-M [32], GoogLeNet [31], and ResNet [25]. The backbone CNN model is partitioned into two parts by inserting the `ViewPool` layer, leading to the formation of the three modules. The `ViewPool` layer itself composes the feature aggregation module, while the feature extraction module includes all layers preceding the `ViewPool` layer, and the classification module includes all layers following the `ViewPool` layer. Let $\mathcal{L}^E$, $\mathcal{L}^A$, and $\mathcal{L}^C$ denote the layer sets of the feature extraction module, feature aggregation module, and classification module, respectively. The specific set of layers included in the feature extraction module (e.g., $\mathcal{L}^E$) and classification module (e.g., $\mathcal{L}^C$) may change based on the placement of the `ViewPool` layer [32]. Typically, the feature extraction module primarily consists of convolution (`CONV`) layers and pooling (e.g., `MaxPool`) layers, while the classification module is primarily composed of fully-connected (`FC`) layers. However, depending on the location of the `ViewPool` layer, the feature extraction module may also contain `FC` layers, and the classification module may also include `CONV` and `MaxPool` layers.

The `CONV` layers extract features from the input feature maps[1] by convolving them with a set of filters to produce output feature maps. For each `CONV` layer $l \in \mathcal{L}^E \cup \mathcal{L}^C$, we define tuples (H_l^I, W_l^I, D_l^I) and (H_l^O, W_l^O, D_l^O) to represent the height, width, and depth dimensions of the input and output feature maps of layer l respectively. The output feature maps at layer l are obtained by convolving the input feature maps with D_l^O filters. The

[1] In the first `CONV` layer, the input image is considered as the input feature maps.

filters at layer l may or may not have the same size depending on the specific architecture employed in the CNN [68], [69]. We assume that filters in layer l have the same dimension, which is denoted as $(h_l\ , h_l, D_l^I)$, without loss of generality. Applying one filter to the input feature maps results in creation of one depth slice in the output feature map. To achieve this, each filter i $(1 \leq i \leq D_l^O)$ at layer l is slid over the input feature maps, computing the dot product between the filter and the portion of the input feature maps that is currently being overlapped by the filter, and then moving the filter to the next region and repeating the process. To compute one of $H_l^O W_l^O$ data elements in the i-th depth slice of the output feature map, it requires $(h_l)^2 D_l^I$ multiplications and $(h_l)^2 D_l^I - 1$ additions.

The `FC` layers map the learned features from `CONV` layers to the output classes. For each `FC` layer $l \in \mathcal{L}^E \cup \mathcal{L}^C$, let X_l^I and X_l^O denote the input and output dimensions of layer l. X_l^I data elements in the output of the previous layer are connected to every neuron in layer l, resulting in X_l^O output data elements. Each connection between the neurons has weight and bias associated with it. The computation of one of X_l^O data elements in the output involves the calculation of a weighted sum of the input. Specifically, X_l^I data elements in the input are multiplied with their corresponding weights, and the resulting products are summed together with a bias term to obtain the weighted sum. Therefore, to generate one output data element at layer l, X_l^I multiplications and X_l^I additions are required.

Activation layers with different activation functions (e.g., `sigmoid`, `ReLu`, `Tanh`, etc) are usually applied after `CONV` and `FC` layers to introduce non-linearity to the model. A special activation layer with `Softmax` function as activation function is applied to the last `FC` layer to produce a probability distribution over all possible classes. The pooling layers in feature extraction module, such as `MaxPool` layers, down-sample the output of the

preceding layer by selecting the maximum value from a region in the input, reducing the data dimensionality and matching the size of output with the input size of the subsequent layer. In contrast, the pooling layer in feature aggregation module, such as the `ViewPool` layer, usually apply element-wise maximum operation or arithmetic mean operation to combine features from multiple views and form a single, compact feature representation of the 3D object [32], [33]. Normalization layers normalize the features before passing them on as the input of the next layer to reduce internal covariate shift. To simplify the analysis, we omit the computing cost associated with these layers, as they generally require significantly less computation compared to the `CONV` and `FC` layers [69], [70], [71].

Let v_l denote the number of output data elements at layer $l \in \mathcal{L}^E \cup \mathcal{L}^A \cup \mathcal{L}^C$, given by

$$v_l = \begin{cases} H_l^O W_l^O D_l^O, & \text{if layer } l \text{ is } \texttt{CONV} \text{ layer} \\ X_l^O, & \text{if layer } l \text{ is } \texttt{FC} \text{ layer.} \end{cases} \tag{3.2}$$

The number of floating-point operations, including both multiplications and additions, for computing one output data element at layer $l \in \mathcal{L}^E \cup \mathcal{L}^A \cup \mathcal{L}^C$ is denoted as π_l and can be expressed as

$$\pi_l = \begin{cases} 2(h_l)^2 D_l^I - 1, & \text{if layer } l \text{ is a } \texttt{CONV} \text{ layer} \\ 2X_l^I, & \text{if layer } l \text{ is a } \texttt{FC} \text{ layer} \\ 0 & \text{otherwise.} \end{cases} \tag{3.3}$$

3.3 Computing Model

Let ω denote the computation intensity representing the number of CPU cycles required to perform one floating-point operation. The computation workload (in CPU cycles) for any layer $l \in \mathcal{L}^E \cup \mathcal{L}^A \cup \mathcal{L}^C$, denoted as λ_l, can be described as $\lambda_l = \omega v_l \pi_l$. The computing capability of AV $k \in \mathcal{K}_s(t)$ is described by its maximum CPU frequency F_k (in CPU cycles per second), and the CPU frequency available for the cooperative classification task at time slot t is denoted by $f_k(t)$. Due to the varying computation workloads of each AV, the exact computing resource each AV can provide for cooperative classification is based on its current workload and resource availability at time t, and thus we have $f_k(t) \in [0, F_k]$. We assume that $f_k(t)$ remains static during each time slot t.

In every time slot, it is required for all AVs in subgroup $\mathcal{K}_s(t)$, including both helpers and the aggregator, to execute feature extraction module. This entails generating a computation workload of $\sum_{l \in \mathcal{L}^E} \lambda_l$ and consuming computing resources $f_k(t)$ for each AV in subgroup. Let $d_k^E(t)$ denote the computing delay for executing feature extraction module at AV k at time slot t, given by

$$d_k^E(t) = \frac{\sum_{l \in \mathcal{L}^E} \lambda_l}{f_k(t)}, \quad \forall k \in \mathcal{K}_s(t). \tag{3.4}$$

The execution of feature extraction module is initiated simultaneously by all AVs in the subgroup. Therefore, the total computing delay experienced by the subgroup for executing feature extraction module at time slot t, denoted by $d^E(t)$, is determined as the maximum

among the feature extraction delays of individual AVs. This is expressed as

$$d^E(t) = \max_{k \in \mathcal{K}_s(t)} d_k^E(t). \tag{3.5}$$

In addition to executing the feature extraction module, aggregator $k_A(t)$ of the subgroup is responsible for executing the feature aggregation module and the classification module. The execution of these two modules results in an additional computation workload of $\sum_{l \in \mathcal{L}^A \cup \mathcal{L}^C} \lambda_l$ for the aggregator. Let $d^C(t)$ denote the computing delay for executing the feature aggregation and classification modules at aggregator $k_A(t)$ at time slot t, given by

$$d^C(t) = \frac{\sum_{l \in \mathcal{L}^A \cup \mathcal{L}^C} \lambda_l}{f_{k_A(t)}(t)}. \tag{3.6}$$

To assess the computation demand of a subgroup in executing cooperative classification, we take into account both the computation workload generated and computing resources consumed by each AV within the subgroup. Specifically, we define the computation demand of an AV, $k \in \mathcal{K}_s(t)$, for executing cooperative classification at time slot t, denoted as $\zeta_k(t)$, as a function of computation workload generated by the AV and computing resources consumed by the AV, given by

$$\zeta_k(t) = \begin{cases} \kappa (f_k(t))^2 \sum_{l \in \mathcal{L}^E} \lambda_l, & \forall k \in \mathcal{K}_s^{(h)}(t) \\ \kappa (f_k(t))^2 \sum_{l \in \mathcal{L}^E \cup \mathcal{L}^A \cup \mathcal{L}^C} \lambda_l, & \forall k \in \{k_A(t)\}. \end{cases} \tag{3.7}$$

This equation has the physical interpretation of computing energy consumption, specifically when parameter κ represents the energy efficiency coefficient [72] [73]. Essentially,

as the consumption of computing resources increases or more computation workload is generated for a computation task, the computation demand on an AV to execute the task also increases, leading to a reduction in task computing efficiency. Building upon this understanding, the total computation demand required by a subgroup $\mathcal{K}_s(t)$ to execute cooperative classification task at time slot t, denoted as $\zeta_{\mathcal{K}_s(t)}$, can be expressed as

$$\zeta_{\mathcal{K}_s(t)} = \sum_{k \in \mathcal{K}_s(t)} \zeta_k(t). \tag{3.8}$$

3.4 Communication Model

Each helper, $k \in \mathcal{K}_s^{(h)}(t)$, is required to transmit its extracted feature maps to aggregator $k_A(t)$ after the execution of feature extraction module. For simplicity, we assume that the feature transmission process of helpers takes place after all AVs have completed feature extraction. Let $\hat{l} \in \mathcal{L}^E$ denote the last layer of the feature extraction module. This layer produces feature maps with $v_{\hat{l}}$ data elements that need to be transmitted. $v_{\hat{l}}$ can be computed using 3.2, if layer $\hat{l}$ is `CONV` or `FC` layer. For other types of layer, the output of layer $\hat{l}$ might have different number of data elements depending on the input data elements and specific operation performed in this layer. For example, if layer $\hat{l}$ is a `MaxPool` layer, then $v_{\hat{l}}$ is the number of data elements in the feature maps generated by the previous `CONV` layer after down-sampling.

Let δ denote the number of bits required to represent a floating-point number. The data size (in bit) of the data to be transmitted is denoted as ψ and can be expressed as $\psi = \delta v_{\hat{l}}$ before compression. Note that the uncompressed data size ψ is independent of the

choice of helper and remains constant across all helpers. This is because a same multi-view classification network is used across all helpers, which results in the feature extraction module outputting the same number of data element, $v_{\hat{l}}$, for all AVs. The data size of the compressed feature maps may differ due to difference in the data they contain. Moreover, the different choice of compression algorithms can also impact the size of the compressed data. Here, we assume that a universal lossless compression algorithm is employed by all helpers for data compression. Hence, the data size (in bit) of compressed data for helper $k \in \mathcal{K}_s^{(h)}(t)$ at time slot t, denoted as $\psi_k(t)$, depends solely on the feature data extracted by helper k.

During feature transmission, multiple helpers need to transmit their extracted feature maps to the aggregator. However, due to each AV having only one transceiver for data message exchange, it is not possible for multiple helpers to transmit their feature maps to a single aggregator simultaneously. To prevent message collision and reduce communication overhead incurred from signaling, the helpers transmit their feature maps to the aggregator sequentially, one after another, utilizing the total available radio spectrum bandwidth B. Based on the Shannon channel capacity formula, the transmission delay for transmitting feature maps of helper $k \in \mathcal{K}_s^{(h)}(t)$ to aggregator $k_A(t)$ at time slot t, denoted by $d_k^T(t)$, is given by

$$d_k^T(t) = \frac{\psi_k(t)}{B \log_2\left(1 + \frac{p g_k(t)}{\sigma^2}\right)}, \quad \forall k \in \mathcal{K}_s^{(h)}(t) \tag{3.9}$$

where p denotes the fixed transmission power for all helpers, $g_k(t)$ denotes the transmission power gain between the helper k and the aggregator $k_A(t)$ at time slot t, and σ^2 represents the received noise power. We assume that the transmission power gain $g_k(t)$ remains static

during feature transmission. The aggregator can only proceed to feature aggregation phase when it has received feature maps from all helpers. Thus, the total feature transmission delay incurred by helpers at time slot t, denoted by $d^T(t)$, is given by

$$d^T(t) = \sum_{k \in \mathcal{K}_s^{(h)}(t)} d_k^T(t). \tag{3.10}$$

As the classification result is represented by a probability vector that is considerably smaller in data size compared to the feature maps, we disregard the transmission delay for the aggregator transmitting the classification result back to all AVs in group $\mathcal{K}$.

Then the sum delay $d_{\mathcal{K}_s(t)}$ experienced by subgroup $\mathcal{K}_s(t)$ at time slot t for performing cooperative classification is given by

$$d_{\mathcal{K}_s(t)} = d^E(t) + d^T(t) + d^C(t). \tag{3.11}$$

3.5 Cooperative Classification Accuracy

Due to the inherent randomness in DNNs and the dynamic nature of a driving environment, the cooperative classification accuracy of a subgroup cannot be accurately estimated overtime. The exact cooperative classification accuracy that a subgroup can obtained at a particular time slot can only be known upon the completion of cooperative classification at that time slot. Define $\phi_{\mathcal{K}_s(t)} \in [0, 1]$ as the cooperative classification accuracy obtained by a subgroup $\mathcal{K}_s(t) \subseteq \mathcal{K}$ at time slot t. This accuracy, measured by true class probability in the probability vector, is generated by the multi-view classification network and reflects

the level of confidence in the classification decision made by the subgroup. In reality, the true class, in the absence of definitive ground truth, can be estimated through a consensus mechanism, wherein the classification result agreed upon by the majority of AVs in the group is taken as the most probable true class. The estimated true class serves as a reference for calibrating and verifying the outcomes of cooperative classification. In addition, as the environment around AVs and the object of interest continually evolves, cooperative classification accuracy $\phi_{\mathcal{K}_s(t)}$ achieved by a subgroup varies across different time slots.

3.6 Summary

In this chapter, we provide a system model for the considered autonomous driving scenario and the proposed cooperative classification scheme. The system model comprises a task model that describes the operation and workload associated with executing cooperative classification task, a computing model that describes the computing delay involved in executing the feature extraction and classification modules, and a communication model that describes the communication delay associated with feature transmission. Furthermore, the acquisition of cooperative classification accuracy for each subgroup is discussed, and the associated notation is provided.

Chapter 4

Problem Formulation and Solution

4.1 Problem Formulation

In our proposed cooperative classification scheme, the group head is responsible for making sequential decisions regarding the selection of a subgroup of AVs ($\mathcal{K}_s(t) \in \mathcal{K}$), which involves determining composition $\boldsymbol{\alpha}(t)$ and role assignment $\boldsymbol{\gamma}(t)$ of the subgroup at each time slot t for cooperative classification. The objective is to minimize the total computation demand of group $\mathcal{K}$ for cooperative classifications, while adhering to accuracy

and delay constraints throughout T time slots. The problem is formulated as

$$\textbf{(P1)} \quad \min_{\{\boldsymbol{\alpha}(t),\boldsymbol{\gamma}(t),\forall t\}} \quad \sum_{t=1}^{T} \zeta_{\mathcal{K}_s(t)} \tag{4.1}$$

$$\text{s.t.} \quad d_{\mathcal{K}_s(t)} \leq \tau, \quad \forall t \tag{4.2}$$

$$\phi_{\mathcal{K}_s(t)} \geq \Theta, \quad \forall t \tag{4.3}$$

$$\alpha_k(t) \in \{0,1\}, \quad \forall t \text{ and } \forall k \in \mathcal{K} \tag{4.4}$$

$$\gamma_k(t) \in \{0,1\}, \quad \forall t \text{ and } \forall k \in \mathcal{K} \tag{4.5}$$

$$\gamma_k(t) \leq \alpha_k(t), \quad \forall k \in \mathcal{K} \tag{4.6}$$

$$\sum_{k\in\mathcal{K}} \gamma_k(t) = 1. \tag{4.7}$$

Constraint 4.2 guarantees that the total delay incurred from each cooperative classification cannot exceed the maximum tolerable delay requirement τ. Constraint 4.3 guarantees that the accuracy of cooperative classification obtained by the selected subgroup $\mathcal{K}_s(t)$ at any time slot t should satisfy the minimum classification accuracy requirement Θ. Constraint 4.4 represents that each AV in the group $\mathcal{K}$ can either be selected to become a member of the subgroup or become a freeloader. Constraint 4.5 represents that each AV in the subgroup $\mathcal{K}_s(t)$ can either serve as aggregator or helper. Constraint 4.6 ensures that the roles of aggregator and helper can only be assigned to AVs that are part of the subgroup, preventing the selection of AVs outside the subgroup for these roles. Constraint 4.7 ensures that only one AV in group $\mathcal{K}$ can serve as aggregator at any given time slot for cooperative classification.

To solve the subgroup scheduling optimization problem (P1) with conventional opti-

mization techniques, prior knowledge on cooperative classification accuracy, delay, and available computing resource of each candidate subgroup is necessary. However, as discussed in Section 1.4, the acquisition or estimation of this information presents significant challenges. To address the challenges posed by solving (P1), we adopt an online learning framework which does not demand knowledge of this information and, instead, explores and learns necessary information from the operating environment while simultaneously using the gathered information to guide decision-making process. Specifically, we reformulate (P1) as a variant of standard MAB problem, known as the MAB with cost subsidy problem, and utilize tools from MAB theory to devise solution.

In a standard MAB problem, a gambler is presented with a set of $|\mathcal{A}|$ slot machines known as bandits. Each bandit $a \in \mathcal{A}$ has an arm that, upon being pulled, generates a random reward r_a drawn independently from a fixed, but unknown distribution $\mathcal{F}_a$ with mean μ_a. At each time slot t, the gambler pulls the arm of a bandit $a_t \in \mathcal{A}$ and receives a reward $r_{t,a}$ drawn from distribution $\mathcal{F}_{a_t}$ with mean μ_{a_t} [74], [75]. The gambler's objective is to make sequential decisions regarding which arms to pull at each time slot over a given time horizon T to maximize the cumulative rewards. As the gambler lacks prior knowledge of the reward distribution associated with each arm, the goal of MAB algorithms is to learn about the reward distribution associated with each arm by observing historical reward outcomes, and utilize learned knowledge to make the most beneficial decisions for subsequent arm selections. In real-world applications of MAB, in addition to rewards, costs are also associated with selecting an arm, with high-performing arms tend to incur higher costs. To account for the presence of costs alongside rewards, modifications need to be applied to the standard MAB problem framework to incorporate the cost associated

with each arm and optimize the trade-off between cumulative rewards and costs.

Common approaches to balance the trade-off between reward and cost involve employing simple arithmetic operations, such as weighted sum or division, to incorporate reward and cost metrics as a modified optimization objective in standard MAB problem formulation [76]. However, it is important to note that this modification may not always be meaningful, particularly when the reward and cost metrics have different scales and correspond to different quantities. A variant of the standard MAB problem known as MAB with cost subsidy has been introduced to handle the subtlety in balancing reward-cost trade-off [75]. In the MAB with cost subsidy problem setting, the reward $r_{t,a}$ and cost $c_{t,a}$ received by arm a at time slot t are drawn from two unknown distributions $\mathcal{F}_a^r$ and $\mathcal{F}_a^c$ with mean $\mu_{r,a}$ and $\mu_{c,a}$ respectively. Let a_r^* denotes the arm with the highest mean reward among all available arms, i.e., $a_r^* = \arg\max_{a \in \mathcal{A}} \mu_{r,a}$. The mean reward of this arm is represented as μ_{r,a_r^*}. To manage cost, the gambler is willing to accept a loss from highest reward and receive only a fraction of the highest mean reward, referred to as the smallest tolerable reward, represented as $(1-\rho)\mu_{r,a_r^*}$, where ρ denotes the user-specified cost subsidy factor that controls the value of the smallest tolerable reward. Arms whose mean reward exceeds $(1-\rho)$ factor of the highest mean reward are collected in a feasible arm set, given by

$$\mathcal{A}^* = \{a \in \mathcal{A} \mid \mu_{r,a} > (1-\rho)\mu_{r,a_r^*}\}. \tag{4.8}$$

The goal of MAB with cost subsidy algorithm is to learn the cheapest arm whose mean reward is at least as large as the smallest tolerable reward, i.e., identifying the optimal arm $a^* = \arg\min_{a \in \mathcal{A}^*} \mu_{c,a}$. The mean cost of the optimal arm is represented by μ_{c,a^*}. In other

words, instead of aiming to identify and select the arm with highest mean reward, the goal here is to identify and maximize the number of selection of the cheapest arm from feasible arm set $\mathcal{A}^*$ to provide tolerable cumulative rewards and minimize cumulative cost over T time slots. This entails selecting the arm that minimize both the reward regret, represented by $\max\{(1-\rho)\mu_{r,a_r^*} - \mu_{r,a_t}, 0\}$, and the cost regret, represented by $\max\{\mu_{c,a_t} - \mu_{c,a^*}, 0\}$, at each time slot t. By using two notions of regret, namely reward regret $\mathcal{R}_r(\cdot)$ and cost regret $\mathcal{R}_c(\cdot)$, such goal can be formally expressed as

$$\begin{aligned}
&\min_{a_1,\ldots,a_T} \left[\mathcal{R}_r(T,\rho,\boldsymbol{\mu}_r,\boldsymbol{\mu}_c) + \mathcal{R}_c(T,\rho,\boldsymbol{\mu}_r,\boldsymbol{\mu}_c)\right] \\
&\qquad = \mathbb{E}\left[\sum_{t=1}^{T}\left(\max\{(1-\rho)\mu_{r,a_r^*} - \mu_{r,a_t}, 0\}\right) + \sum_{t=1}^{T}\left(\max\{\mu_{c,a_t} - \mu_{c,a^*}, 0\}\right)\right],
\end{aligned} \tag{4.9}$$

where vectors $\boldsymbol{\mu}_r = (\mu_{r,1}, \ldots, \mu_{r,|\mathcal{A}|})$ and $\boldsymbol{\mu}_c = (\mu_{c,1}, \ldots, \mu_{c,|\mathcal{A}|})$ represent the mean reward and cost associated with all arms in arm set $\mathcal{A}$. The expectation is taken over the randomness involved in selecting arm a_t according to the designed arm selection algorithm.

(P1) can be effectively considered as a cost-subsidized MAB problem. In our scenario, the group head plays the role of the gambler, which makes sequential decisions regarding the selection of AV subgroups for cooperative classifications. The arm a_t corresponds to the scheduled subgroup for cooperative classification at time slot t, which is represented as tuple $(\mathcal{K}_s(t), k_A(t))$ with $\mathcal{K}_s(t)$ representing the member composition of the subgroup and $k_A(t)$ representing the designated aggregator for the subgroup. The arm set, $\mathcal{A}$, corresponds to the set of all possible candidate subgroups, which is expressed as a collection of tuples $(\mathcal{K}_s, k_A)$

$$\mathcal{A} = \{(\mathcal{K}_s, k_A) \mid \mathcal{K}_s \in 2^{\mathcal{K}} \setminus \{\emptyset\}, k_A \in \mathcal{K}_s\}, \tag{4.10}$$

where $2^{\mathcal{K}}$ is the power set of group $\mathcal{K}$, which encompasses all possible subsets of group $\mathcal{K}$. The set $\mathcal{A}$ comprises all possible combinations of member composition $\mathcal{K}_s$ and aggregator k_A for group $\mathcal{K}$, constituting all the available arms for pulling at each time slot. Upon the selection of a subgroup a_t at time slot t, a reward, r_{t,a_t}, associated with quality of service (QoS), is revealed, and concurrently, a cost, c_{t,a_t}, related to computation demand is incurred. The objective of scheduling subgroups over the tracking period to achieve desired classification accuracy and delay while minimizing computation demand can be viewed as selecting arms over T time slots to provide tolerable cumulative QoS reward and minimize cumulative computational cost. Throughout the remainder of this thesis, we use a to represent an instance of subgroup tuple from the subgroup set $\mathcal{A}$, i.e. $a = (\mathcal{K}_s, k_A)$, $\forall a \in \mathcal{A}$.

4.2 Problem Solution

4.2.1 Overview

In the conventional approach to solving the MAB with cost subsidy problem, a strategy, known as *explore-then-commit*, is employed. This strategy involves two phases, encompassing a pure exploration phase and a exploitation phase. In the pure exploration phase, each arm is selected for a predefined number of rounds, from which reward and cost data are collected and the mean reward and mean cost associated with each arm are estimated. The strategy then advances to the exploitation phase, wherein a feasible set of arms is constructed based on the upper and lower confidence bounds on the reward of each arm, from which the arm with the least cost is selected [75]. To apply *explore-then-commit* strategy in

our context, all possible subgroup configurations within the group will need to be explored for multiple times. This strategy functions effectively when the group comprises only a limited number of AVs. However, as the number of vehicles within the group grows, the number of possible subgroup configurations can escalate significantly. For example, a group comprising just 5 AVs gives rise to 80 possible subgroup configurations, whereas a group encompassing 10 AVs results in a substantial 5120 potential subgroup configurations. As a consequence, the duration of the pure exploration phase becomes considerably extended, hindering the ability of *explore-then-commit* strategy to identify the optimal subgroup within a limited period of time. In order to apply *explore-then-commit* strategy within our context, additional measures need to be taken to reduce the number of subgroups to be explored during the pure exploration phase.

Despite the presence of randomness and uncertainty in the problem, certain patterns can be observed and harnessed toward the objective of reducing subgroup exploration space. For example, a subgroup consisting of AVs with clear and close views of the object generally provides more valuable information in cooperative classification than a subgroup of AVs with obstructed and distant views of the object. While the exact cooperative classification accuracy of these two subgroups cannot be determined, it is expected that the former subgroup, on average, can yield more accurate cooperative classification result. Hence, for subgroups of equal size, if the subgroups possess superior viewing conditions fails to satisfy the classification accuracy requirement, it becomes unlikely for subgroups possessing less favorable viewing conditions to satisfy the same classification accuracy requirement. By only exploring and examining the cooperative classification accuracy performance of subgroups with superior viewing conditions, we facilitate the satisfaction of classification

accuracy requirement and bypass the need to explore all subgroups of equal size. On the other hand, Upon observing a subgroup of a smaller size achieves the desired classification accuracy, it often indicates that larger subgroups containing vehicles from this smaller group are likely to attain the desirable accuracy threshold as well. Nonetheless, focusing on smaller subgroups is beneficial for our optimization objective, as fewer vehicles in a subgroup result in reduced classification delays and decreased computation demand due to fewer feature transmission and extraction. Therefore, when smaller-sized subgroups demonstrate the capacity to attain the accuracy threshold, further exploration on larger-sized subgroups can be avoided.

To realize this idea, we begin by defining criteria and crafting tools to assess the viewing condition of subgroups for cooperative classification. Employing the criteria and tools, an algorithm is devised to identify subgroups with superior viewing conditions from all subgroups of the same size. The algorithm is subsequently used by another algorithm to reduce the subgroup exploration space. In the process of reducing subgroup exploration space, a coarse-grained exploration is conducted on the subgroups with superior viewing conditions, through which their suitability in achieving our optimization objective is assessed. Unsuitable subgroups, along with those of the same size, are discarded, while the suitable ones are retained and returned. Finally, the *explore-then-commit* strategy is applied to perform fine-grained exploration on the subgroups returned by the preceding algorithm, from which the most suitable subgroups that align with our optimization objective are scheduled for cooperative classifications.

4.2.2 Viewing Condition Assessment

In assessing the viewing conditions of a subgroup of vehicles for cooperative classification, several criteria and their impacts have been identified.

Distance: The object detection accuracy has been experimentally verified to be influenced by the distance, d, between the observer and the object of interest [77]. The decline of detection performance for very close object stems from the fact that the objects at closer ranges are usually more truncated, resulting in only partial observation of the object. The partial observability can subsequently hinder the accurate classification of the object due to the absence of complete features. On the other hand, when the vehicle is too far away from the object, the object appears too small in the image, which affects the resolution of image and loses fine-grained information about the object, leading to decreased classification accuracy [78], [79]. As a result, there exists an interval of observation distance, neither too close nor too distant from the object, within which vehicles can capture valuable and high-resolution data about the object, thereby improving the overall classification performance. Based on these insights, we develop a distance-based scoring function to reflect the value of the information that each vehicle can contribute to the cooperative classification process. Such function is defined as

$$S_k^{(D)} = \begin{cases} \exp\left(-\frac{(d-D_{\text{near}})^2}{2\Gamma_{\text{near}}^2}\right) & \text{if } d \leq D_{\text{near}} \\ 1 & \text{if } D_{\text{near}} < d \leq D_{\text{far}} \\ \exp\left(-\frac{(d-D_{\text{far}})^2}{2\Gamma_{\text{far}}^2}\right) & \text{if } d > D_{\text{far}} \end{cases}, \qquad \forall k \in \mathcal{K} \tag{4.11}$$

where $S_k^{(D)}$ represents a distance-based score for AV k, D_{near} and D_{far} represent the close and far distance threshold from which the object classification performance starts declining. The decay rate at close range and distant range are controlled by Γ_{near} and Γ_{far}. This score function penalizes vehicles which are too close or too distant to the object of interest and favors vehicles which are positioned within the optimal observation range, i.e., $[D_{\text{near}}, D_{\text{far}}]$.

Viewing perspectives: Multi-view classification networks, i.e., MVCNN, value discriminative information more than redundant information [31]. Consequently, diversified viewing perspectives amplify the efficiency of multi-view object classification. In contrast, similar viewing perspectives tend to yield redundant and overlapping data, which undermines the cooperative classification performance. In the considered scenario, diverse viewing perspectives of the object of interest are primarily attained by allowing vehicles in different lanes to participate in the cooperative classification process. Conversely, having vehicles within the same lane to perform cooperative classification result in providing overlapping or similar perspectives of the object.

Occlusion: Clear views provide more information about the object of interest than views hindered by obstructions. While we recognize the significance of occlusion in assessing the viewing conditions of vehicles, the specific relationships between vehicles with respect to occlusion remain uncertain. Specifically, we cannot precisely determine the degree to which the object of interest is obscured by other vehicles or other traffic participants in the environment. This ambiguity stems from the lack of complete information of everything on the road, such as the shape of vehicles, road and visibility conditions [67], [80]. Nonetheless,

with the limited information available, such as the positions of vehicles and the width and length of vehicles, we can estimate whether a vehicle's view of the object of interest is obstructed. We do this by checking if other vehicles are located in the direct line of sight between the observing vehicle and the object of interest, from which we can estimate which vehicles are likely having an obstructed view to the object of interest, and which vehicles are likely having clear views.

Given the position of a source vehicle, denoted as (x_s, y_s), and the object of interest, represented by (x_o, y_o), the line of sight connecting them can be expressed as parametric equation of a line

$$x = x_s + t(x_o - x_s) = x_s + t\Delta x, \quad 0 \leq t \leq 1 \tag{4.12}$$

$$y = y_s + t(y_o - y_s) = y_s + t\Delta y, \quad 0 \leq t \leq 1 \tag{4.13}$$

where $t = 0$ and $t = 1$ represent the starting and ending points of the line segment. Given the width w and length l of a vehicle at position, (x_k, y_k), the vehicle can be represented by a bounding box defined by its bottom-left vertex, (x_B, y_B), and its top-right vertex, (x_T, y_T). The coordinates of these vertices are given by

$$x_B = x_k - \frac{l}{2}, \quad y_B = y_k - \frac{w}{2}, \tag{4.14}$$

$$x_T = x_k + \frac{l}{2}, \quad y_T = y_k + \frac{w}{2}. \tag{4.15}$$

The estimation of obstructed views can be considered as a problem of checking whether a line, i.e., the line of sight, intersects with a rectangle, i.e., vehicle bounding box. Such

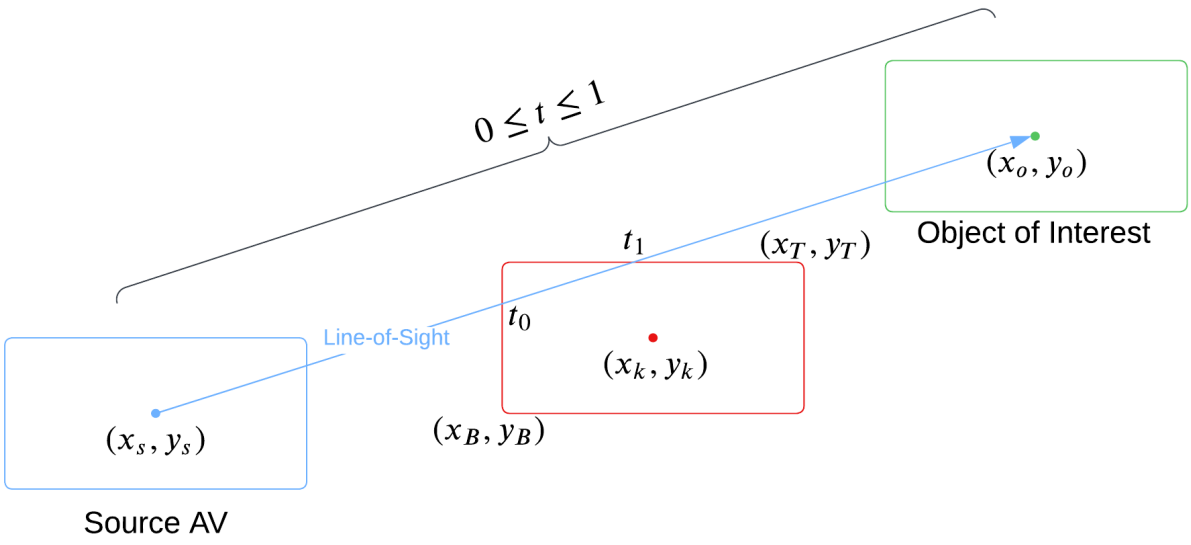


Figure 4.1: An illustration of parameters within the occlusion estimation algorithm.

problem has been extensively studied in computer graphic field and can be efficiently solved by an algorithm known as the Liang–Barsky algorithm [81].

Algorithm 1 summarizes the occlusion estimation algorithm derived from the Liang–Barsky algorithm. In essence, the algorithm determines if the line of sight between the source vehicle and the object of interest is blocked by the bounding box of another vehicle, as depicted in Figure 4.1, by checking if any points of the line is in the bounding box. A point is in the bounding box if the following conditions are satisfied

$$x_B \leq x_s + t\Delta x \leq x_T \tag{4.16}$$

$$y_B \leq y_s + t\Delta y \leq y_T. \tag{4.17}$$

This can be re-expressed by the following four inequalities, one for each edge of the bound-

Algorithm 1 Occlusion Estimation with Liang-Barsky Algorithm

Input: Source vehicle position: (x_s, y_s), Object of interest position: (x_o, y_o), Position, width and length of the vehicle being assessed: (x_k, y_k), w, l
Output: True/False on whether vehicle at position (x_k, y_k) is positioned on the line of sight between source vehicle at position (x_s, y_s) and the object of interest at position (x_o, y_o)
Initialization: Determine the bounding box coordinates, (x_B, y_B) and (x_T, y_T), of the vehicle being assessed based on Equations 4.14 and 4.15
Define variables for estimation criteria: $p = [-(x_o - x_s), x_o - x_s, -(y_o - y_s), y_o - y_s]$, $q = [x_s - x_B, x_T - x_s, y_s - y_B, y_T - y_s]$
Initialize the starting and ending points of the line-of-sight segment that is clipped by the vehicle bounding box: $t_0 = 0$, $t_1 = 1$
for $i = 0 \rightarrow 3$ **do**
  if $p_i == 0$ **then**
    if $q_i < 0$ **then**
      return False
    end if
    continue
  end if
  Update the parameter corresponding to the intersection point between the line of sight and the current edge: $t \leftarrow \frac{q_i}{p_i}$
  if $p_i < 0$ **then**
    $t_0 \leftarrow \max(t_0, t)$
  else
    $t_1 \leftarrow \min(t_1, t)$
  end if
  if $t_0 > t_1$ **then**
    return False
  end if
end for
return True

ing box

$$tp_i \leq q_i, \quad i = 1, 2, 3, 4 \tag{4.18}$$

where

$$\begin{aligned} p_1 &= -\Delta x & q_1 &= x_s - x_B & \text{(left edge)} \\ p_2 &= \Delta x & q_2 &= x_T - x_s & \text{(right edge)} \\ p_3 &= -\Delta y & q_3 &= y_s - y_B & \text{(bottom edge)} \\ p_4 &= \Delta y & q_4 &= y_T - y_s & \text{(top edge).} \end{aligned} \tag{4.19}$$

The value of p_i indicates the orientation of the line of sight concerning the edge of the bounding box. A negative p_i value suggests the line of sight moves from outside towards the inside of edge i, a positive p_i value implies it moves from inside to outside, while $p_i = 0$ denotes that the line of sight is parallel to edge i. The value of q_i signifies the distance from the starting point of the line of sight to edge i of the bounding box. The algorithm iterates over each edge of the estimated vehicle bounding box. In lines 5 to 10, the algorithm evaluates whether the line of sight of the source vehicle runs parallel to and remains outside the bounding box. If this condition holds, indicating the line of sight does not intersect with the bounding box of the vehicle being assessed, it returns false and moves on to the next edge. In lines 12 to 19, the parameters t_0 and t_1, defining the starting and ending points of the line segment of the line of sight clipped by the bounding box, are computed for edges not parallel to the line of sight. If t_0 exceeds t_1, it suggests that the line is completely outside the vehicle bounding box, leading to a return of false. Conversely, if this condition is not met, the algorithm returns true, signifying a potential obstruction of the view by the vehicle at the position (x_k, y_k).

4.2.3 Identification of Superior Subgroup

Algorithm 2 outlines the systematic procedure we employ to determine, for any feasible subgroup size, which subgroups have superior viewing conditions based on the insights and tools we have developed above. In essence, the algorithm pinpoints the subgroups comprising vehicles positioned close to the $[D_{\text{near}}, D_{\text{far}}]$ range and possessing clear lines of sight towards the object of interest, marking them as having superior viewing conditions for cooperative classification. Having close and clear views not only present a richer information about the object of interest, but also promote a diversified viewing perspective. This is attributed to the fact that vehicles with close and clear views are commonly positioned at the front of different lanes, providing diverse viewing perspective. Therefore, the algorithm starts by evaluating the line-of-sight statuses of vehicles in group $\mathcal{K}$, categorizing them into vehicles with clear views and vehicles with obstructed views using Algorithm 1. Following the categorization, the AVs with clear views are ranked based on their distance score $S_k^{(D)}$ and stored in the set $\mathcal{V}^{(C)}$. Similarly, AVs with obstructed views are ranked using the same metric and stored in set $\mathcal{V}^{(O)}$. By ranking the AVs with clear views based on the distance metric, we emphasize the identification of vehicles positioned within, or proximate to, the $[D_{\text{near}}, D_{\text{far}}]$ range and possessing clear line of sights. The ranking of AVs in the obstructed category is based on the understanding that, despite obstruction, being located within the optimal observation range can offer a richer visual context of the target object, thereby enhancing the cooperative classification performance. Note that, for both categories, multiple vehicles might share the same rank, indicating they have same line of sight characteristics and same distance score.

Algorithm 2 Proposed Algorithm to Identify Subgroups with Superior Viewing Conditions

Input: Group $\mathcal{K}$, Subgroup size: n
Output: Subgroups of size n with superior viewing conditions
Initialization: Determine AVs with clear views and AVs with obstructed views from group $\mathcal{K}$ using Algorithm 1
Rank AVs with clear views according to their distance-based score $S_k^{(D)}$ and store them in set $\mathcal{V}^{(C)}$
Rank AVs with obstructed views according to their distance-based score $S_k^{(D)}$ and store them in set $\mathcal{V}^{(O)}$
if $1 \leq n \leq |\mathcal{V}^{(C)}|$ **then**
  if multiple vehicles have same rank in $\mathcal{V}^{(C)}$ **then**
    $\mathcal{C}_n(\mathcal{V}^{(C)}) \leftarrow \{\text{all vehicle combinations of size } n \text{ from } \mathcal{V}^{(C)}\}$
    $S_{max}^{(D)} \leftarrow \max_{\mathcal{C} \in \mathcal{C}_n(\mathcal{V}^{(C)})} \sum_{k \in \mathcal{C}} S_k^{(D)}$
    return $\{\mathcal{C} \mid \mathcal{C} \in \mathcal{C}_n(\mathcal{V}^{(C)}) \text{ and } \sum_{k \in \mathcal{C}} S_k^{(D)} = S_{max}^{(D)}\}$
  else
    return $\{k_i \mid k_i \in \mathcal{V}^{(C)} \text{ and } 1 \leq i \leq n\}$
  end if
end if
if $n > |\mathcal{V}^{(C)}|$ **then**
  Compute the number of remaining spots in subgroup: $n_r = n - |\mathcal{V}^{(C)}|$
  if multiple vehicles have same rank in $\mathcal{V}^{(O)}$ **then**
    $\mathcal{C}_{n_r}(\mathcal{V}^{(O)}) \leftarrow \{\text{all vehicle combinations of size } n_r \text{ from } \mathcal{V}^{(O)}\}$
    $S_{max}^{(D)} \leftarrow \max_{\mathcal{C} \in \mathcal{C}_{n_r}(\mathcal{V}^{(O)})} \sum_{k \in \mathcal{C}} S_k^{(D)}$
    return $\{\mathcal{V}^{(C)} \cup \mathcal{C} \mid \mathcal{C} \in \mathcal{C}_{n_r}(\mathcal{V}^{(O)}) \text{ and } \sum_{k \in \mathcal{C}} S_k^{(D)} = S_{max}^{(D)}\}$
  else
    return $\left\{\mathcal{V}^{(C)} \cup \{k_i \mid k_i \in \mathcal{V}^{(O)} \text{ and } 1 \leq i \leq n_r\}\right\}$
  end if
end if

In lines 4 to 12, the algorithm handles the scenario where the number of vehicles with clear line of sights toward the object of interest meets or exceeds the required subgroup size. In this case, if multiple vehicles in $\mathcal{V}^{(C)}$ share the same rank, the algorithm finds all combinations of vehicles in $\mathcal{V}^{(C)}$ with a size equal to the subgroup size n, represented as $\mathcal{C}_n(\mathcal{V}^{(C)})$, computes the cumulative distance score of all vehicles in each of the combinations, and determines the maximum cumulative distance score, $S^{(D)}_{max}$, across all vehicle combinations. It then returns all combinations, i.e., subgroups, from $\mathcal{C}_n(\mathcal{V}^{(C)})$ where the sum of distance score for the vehicles within that subgroup equals $S^{(D)}_{max}$. These steps are necessary to ensure the preservation of all subgroups that have equally favorable viewing conditions in terms of distance and clear line of sights, e.g., two subgroups consist of vehicles that have clear views and are positioned within the optimal observation range. As in such situations, solely relying on distance and occlusion metrics does not allow us to ascertain which subgroup would have the best viewing condition among all subgroups consists of vehicles that have clear views and same distance score. Therefore, it is better to retain all these subgroups for further investigation. On the other hand, if no vehicles in $\mathcal{V}^{(C)}$ share the same rank, the algorithm simply returns a subgroup that contains the top n vehicles from $\mathcal{V}^{(C)}$ to match the desirable subgroup size. In lines 13 to 22, the algorithm handles the situation where the required subgroup size n surpasses the total number of vehicles $|\mathcal{V}^{(C)}|$ that possess a clear line of sight to the object of interest. In this case, the algorithm gives precedence to vehicles with clear views and filling the remaining subgroup spots, denoted by n_r, with vehicles having obstructed views. Similar to previous situation, when multiple vehicles in $\mathcal{V}^{(O)}$ hold identical ranks, the algorithm assembles all combinations of vehicles from $\mathcal{V}^{(O)}$ with size n_r, denoted as $\mathcal{C}_{n_r}(\mathcal{V}^{(O)})$, and identifies the vehicle

Algorithm 3 Proposed Subgroup Exploration Space Reducing Algorithm

Input: Group $\mathcal{K}$, Number of exploration: β_1, Classification accuracy requirement: Θ
Output: Subgroups to be fine-grained explored
for size $n = 1, 2, \ldots, |\mathcal{K}|$ **do**
  Identify subgroup(s) with superior viewing conditions and have size n:
  $\mathcal{G}_n \leftarrow$ Invoke Algorithm 2 with input $(\mathcal{K}, n)$
  for all subgroups g in $\mathcal{G}_n$ with distinct member composition **do**
    for $i = 1$ **to** β_1 **do**
      Explore cooperative classification accuracy $\phi_{g,i}$ for subgroup g
    end for
    Compute the empirical mean accuracy for subgroup g: $\hat{\phi}_g \leftarrow \frac{1}{\beta_1} \sum_{i=1}^{\beta_1} \phi_{g,i}$
  end for
  if $\max_{g \in \mathcal{G}_n} \hat{\phi}_g < \Theta$ **then**
    Discard all subgroups with size n.
    continue
  else
    return $\mathcal{G}_n$
  end if
end for

combinations whose cumulative distance score equal to $S_{max}^{(D)}$. It then returns subgroups, each consisting of vehicles from $\mathcal{V}^{(C)}$ combined with those from the identified combinations. Conversely, if no vehicles within $\mathcal{V}^{(O)}$ have the same rank, the algorithm straightforwardly pairs the top n_r vehicles from $\mathcal{V}^{(O)}$ with those in $\mathcal{V}^{(C)}$ and returns the subgroup formed by these vehicles. This algorithm ensures that the returned subgroups comprise a mix of vehicles optimized for both view clarity and proximity to the object of interest.

4.2.4 Subgroup Exploration Space Reduction

Utilizing Algorithm 2, we provide details about the subgroup exploration space reducing algorithm as summarized in Algorithm 3. For every possible subgroup size, the algorithm

calls Algorithm 2 to identify subgroups that have superior viewing conditions and are of size n, denoted as $\mathcal{G}_n$. In line 5 to 10, the algorithm explore and estimate the cooperative classification accuracy performance for subgroups in $\mathcal{G}_n$ that have unique member composition. This is because $\mathcal{G}_n$ may contain multiple subgroups with same member compositions but different role assignment, these subgroups differ by having different cooperative classification delay and computation demand but have same cooperative classification accuracy performance since views provides by AV members are the same. Thus, it is sufficient to explore one of those subgroups in order to determine the cooperative classification accuracy performance for all subgroups with same member composition. Given that subgroups in $\mathcal{G}_n$ represent the optimal viewing conditions for their size, if they fail to meet the classification accuracy standards, it is improbable that other subgroups of equivalent size would fulfill this requirement. Therefore, if the highest empirical mean accuracy, $\hat{\phi}_g$, among all the subgroups g with unique member composition in $\mathcal{G}_n$, is less than the predefined accuracy threshold, Θ, then all subgroups of size n are discarded. Otherwise, the algorithm returns all subgroups in $\mathcal{G}_n$. The underlying logic behind this step is that when a subgroup of relatively smaller size attains the desired classification accuracy, it is reasonable to anticipate that larger subgroups encompassing vehicles from this smaller subgroup would also meet the accuracy requirement. In other words, when smaller-sized subgroups demonstrate capability in meeting accuracy requirement, it also implies the capability of bigger subgroups in meeting the accuracy requirement. However, prioritizing smaller subgroups is advantageous in terms of achieving our optimization objective, as a smaller subgroup size leads to reduced cooperative classification delay and decreased computation workload due to having fewer vehicles involved in feature transmission and feature extraction. Thus, once

smaller-sized subgroups have shown potential in satisfying accuracy requirement, we can skip additional exploration of larger-sized subgroups and focus on determining the specific classification performance of them.

4.2.5 Explore-then-Commit

The resulting subgroups from Algorithm 3 are of same size and possess comparable viewing conditions in terms of distance and line of sight. We know from Algorithm 3 that among the returned subgroups, there exists a subset of subgroups with identical member compositions capable of meeting the accuracy requirement. While subgroups with identical member compositions exhibit same cooperative classification accuracy, they may exhibit differences in cooperative classification delay and computation demand because of distinct role assignments. In additions, subgroups with different member compositions may exhibit differences in cooperative classification accuracy, delay, and computation demand. Therefore, performing fine-grained exploration on returned subgroups is essential to identify potential subgroups which not only meet the accuracy requirement but also have reduced cooperative classification delays and low computational demands. To this end, with reduced subgroup exploration space, we directly utilize *explore-then-commit* strategy from MAB theory to schedule subgroups for the remaining time slots to minimize the total computation demand, while facilitating the satisfaction of classification accuracy and delay requirements. Under the cost-subsidized MAB problem formulation, we define the reward

of selecting subgroup a_t at time slot t, denoted by r_{t,a_t}, as

$$r_{t,a_t} = \begin{cases} \Upsilon, & \text{if } d_{\mathcal{K}_s(t)} \leq \tau \text{ and } \phi_{\mathcal{K}_s(t)} \geq \Theta \\ 0 & \text{otherwise.} \end{cases} \tag{4.20}$$

This reward function assigns the same reward, Υ, to all subgroups that meet the classification accuracy and delay requirements, disregarding any differences in their individual performances beyond the specified requirements. Consequently, the reward function does not incentivize the selection of subgroups that overprovision resource to exceed the accuracy and delay requirements. In fact, allocating excessive resources to further reduce delay or improve accuracy beyond the requirements lead to more computation demand and resource inefficiency, which is undesirable [82]. In addition, we define the cost of selecting subgroup a_t at time slot t as the total computation demand required by the selected subgroup for executing cooperative classification at the time slot, given by $c_{t,a_t} = \zeta_{\mathcal{K}_s(t)}$, where $\zeta_{\mathcal{K}_s(t)}$ is given in Equation (3.8).

Algorithm 4 summarizes the proposed AV subgroup scheduling algorithm. The algorithm takes subgroup set, $\mathcal{A}$, the total number of time slots, T, the subsidy factor, ρ, and the number of exploration per subgroup, β_2, as inputs, which are pre-determined by the group head. It outputs the subgroup a_t to be scheduled in each time slot t for performing cooperative classification. The algorithm is divided into two phases, the pure exploration phase, and the exploitation phase. During the pure exploration phase, every candidate subgroup within the subgroup set $\mathcal{A}$ is explored for β_2 time slots, resulting in a total exploration period of $|\mathcal{A}|\beta_2$ time slots. In each exploration time slot, relevant data regarding

Algorithm 4 Subgroup Scheduling by Explore-then-Commit Strategy

Input: Reduced subgroup exploration space: $\mathcal{A}$, Total number of time slots: T, Cost subsidy factor: ρ, Number of exploration: β_2
Output: Subgroup a_t to schedule in each time slot t
Initialization: Total number of times subgroup a is selected $n_a(t)$: $n_a(1) = 0$, $\forall a \in \mathcal{A}$
Pure exploration phase:
for $t = 1, 2, \ldots, |\mathcal{A}|\beta_2$ **do**
    Select subgroup $a_t = t \mod |\mathcal{A}|$ with configuration $(\mathcal{K}_s(t), k_A(t))$ to perform cooperative classification
    Observe its cooperative classification accuracy $\phi_{\mathcal{K}_s(t)}$, cooperative classification delay $d_{\mathcal{K}_s(t)}$, and computation demand $\zeta_{\mathcal{K}_s(t)}$
    Compute reward r_{t,a_t} and cost c_{t,a_t}
    Update empirical reward and cost mean estimation:
    $\hat{\mu}_{r,a}(t) \leftarrow \frac{1}{n_a(t)} \sum_{i=1}^{t} \mathbb{1}\{a_i = a\} r_{i,a_i}$
    $\hat{\mu}_{c,a}(t) \leftarrow \frac{1}{n_a(t)} \sum_{i=1}^{t} \mathbb{1}\{a_i = a\} c_{i,a_i}$
    Update $n_a(t)$: $n_a(t+1) = n_a(t) + \mathbb{1}\{a_t = a\}$, $\forall a \in \mathcal{A}$
end for
Exploitation phase:
for $t = |\mathcal{A}|\beta_2 + 1, \ldots, T$ **do**
    Compute the UCB and LCB of mean reward estimation, and the LCB of mean cost estimation, for all candidate subgroups:
    $\mu_{r,a}^{UCB}(t) \leftarrow \hat{\mu}_{r,a}(t) + \frac{\sqrt{2 \log T}}{n_a(t)}$, $\forall a \in \mathcal{A}$
    $\mu_{r,a}^{LCB}(t) \leftarrow \hat{\mu}_{r,a}(t) - \frac{\sqrt{2 \log T}}{n_a(t)}$, $\forall a \in \mathcal{A}$
    $\mu_{c,a}^{LCB}(t) \leftarrow \hat{\mu}_{c,a}(t) - \frac{\sqrt{2 \log T}}{n_a(t)}$, $\forall a \in \mathcal{A}$
    Identify the subgroup with the highest LCB for the mean reward estimation:
    $a_r^*(t) = \arg\max_{a \in \mathcal{A}} \mu_{r,a}^{LCB}(t)$
    Identify the candidate subgroups with mean reward estimation exceeding the smallest tolerable reward:
    $\mathcal{A}^*(t) = \{a \in \mathcal{A} \mid \mu_{r,a}^{UCB}(t) > (1-\rho)\mu_{r,a_r^*(t)}^{LCB}(t)\}$
    Select subgroup $a_t = \arg\min_{a \in \mathcal{A}^*(t)} \mu_{c,a}^{LCB}(t)$ to observe reward r_{t,a_t} and cost c_{t,a_t}
    Update empirical reward and cost mean estimation:
    $\hat{\mu}_{r,a}(t) \leftarrow \frac{1}{n_a(t)} \sum_{i=1}^{t} \mathbb{1}\{a_i = a\} r_{i,a_i}$
    $\hat{\mu}_{c,a}(t) \leftarrow \frac{1}{n_a(t)} \sum_{i=1}^{t} \mathbb{1}\{a_i = a\} c_{i,a_i}$
    Update $n_a(t)$: $n_a(t+1) = n_a(t) + \mathbb{1}\{a_t = a\}$, $\forall a \in \mathcal{A}$
end for

the performance metrics of each subgroup, such as cooperative classification accuracy, delay, and computation demand, are collected at the end of the time slot to determine the corresponding reward and cost. The reward and cost are then utilized to update the mean reward and mean cost estimations associated with each subgroup using

$$\hat{\mu}_{r,a}(t) = \frac{1}{n_a(t)} \sum_{i=1}^{t} \mathbb{1}\{a_i = a\} r_{i,a_i} \tag{4.21}$$

$$\hat{\mu}_{c,a}(t) = \frac{1}{n_a(t)} \sum_{i=1}^{t} \mathbb{1}\{a_i = a\} c_{i,a_i}, \tag{4.22}$$

where $n_a(t)$ is the total number of times subgroup a has been selected until time slot t. In essence, we continuously estimate the mean reward and cost for each subgroup by averaging the accumulated rewards and costs received from that particular subgroup up to the current time slot. This update process enables the group head to refine and update its estimation of the mean reward and mean cost associated with each candidate subgroup, i.e., $\hat{\mu}_{r,a}(t)$ & $\hat{\mu}_{c,a}(t)$, which can facilitate decision-making during exploitation phase of the algorithm. Conducting a thorough exploration of all candidate subgroups is essential when subgroups offer similar QoS, e.g., comparable mean rewards, but require considerably different computation demand, e.g., different mean costs, for cooperative classification. Achieving optimal results in terms of minimizing both cost and reward regrets necessitates distinguishing between subgroups with similar mean reward but distinct mean cost. Based on the study in [75], in a worst-case scenario where the reward distributions of arms are highly similar, conducting a number of $\sqrt[3]{\left(\frac{T}{|\mathcal{A}|}\right)^2}$ explorations for each subgroup is sufficient for the effective functioning of exploitation phase. Finally, the counter $n_a(t)$ is updated

according to line 10 of the algorithm.

During the exploitation phase, the feasible subgroup set is constructed, and the optimal subgroup is identified and selected from the set. The construction of the feasible subgroup set is based on the current estimations of rewards, while the selection of the optimal subgroup is determined by the current estimations of costs. To this end, the upper confidence bound (UCB) and lower confidence bound (LCB) of the mean reward estimation, and the LCB of mean cost estimation are computed as

$$\mu_{r,a}^{UCB}(t) = \hat{\mu}_{r,a}(t) + \frac{\sqrt{2\log T}}{n_a(t)} \tag{4.23}$$

$$\mu_{r,a}^{LCB}(t) = \hat{\mu}_{r,a}(t) - \frac{\sqrt{2\log T}}{n_a(t)} \tag{4.24}$$

$$\mu_{c,a}^{LCB}(t) = \hat{\mu}_{c,a}(t) - \frac{\sqrt{2\log T}}{n_a(t)}. \tag{4.25}$$

Here, $\mu_{r,a}^{LCB}(t)$ and $\mu_{r,a}^{UCB}(t)$ provide estimate of the confidence interval within which the true values of the mean reward associated with each subgroup are likely to lie. Specifically, $\frac{\sqrt{2\log T}}{n_a(t)}$ is the confidence radius of the interval, $\mu_{r,a}^{UCB}(t)$ represents the estimated upper bound for the confidence interval of mean reward that subgroup a can provide at time slot t, while $\mu_{r,a}^{LCB}(t)$ and $\mu_{c,a}^{LCB}(t)$ represent the estimated lower bounds for the confidence interval of mean reward and mean cost, respectively, for the same subgroup during that time slot [83]. Based on the UCB and LCB on the mean reward estimation of each subgroup, the subgroup that currently has the highest LCB on the mean reward estimation, i.e., $a_r^*(t)$

$= \arg\max_{a \in \mathcal{A}} \mu_{r,a}^{LCB}(t)$, is identified and a feasible subgroup set can be constructed as

$$\mathcal{A}^*(t) = \{a \in \mathcal{A} \mid \mu_{r,a}^{UCB}(t) > (1-\rho)\mu_{r,a_r^*(t)}^{LCB}(t)\}. \tag{4.26}$$

The feasible subgroup set, $\mathcal{A}^*(t)$, serves as an estimation of the set, $\mathcal{A}^*$, which encompasses all subgroups capable of providing at least the smallest tolerable reward. The construction of feasible subgroup set filters out subgroups whose upper limit of the mean reward estimation fall below a user-specified fraction of the lower limit of the mean reward estimation of the subgroup with the highest mean reward i.e., $(1-\rho)\mu_{r,a_r^*(t)}^{LCB}(t)$. This filtering process ensures that the subgroups in $\mathcal{A}^*(t)$ have a greater probability of providing rewards exceed the smallest tolerable reward. The subsequent step involves selecting the subgroup with the lowest LCB on the mean cost estimation from $\mathcal{A}^*(t)$ to perform cooperative classification, which can be expressed as

$$a_t = \arg\min_{a \in \mathcal{A}^*(t)} \mu_{c,a}^{LCB}(t). \tag{4.27}$$

This subgroup selection strategy indicates our aim to identify the most computing efficient subgroup in $\mathcal{A}^*(t)$ based on the current knowledge. Therefore, this algorithm places greater emphasis on selecting a subgroup that can provide tolerable QoS at a minimal cost, rather than focusing on subgroups that offer superior QoS. Once the subgroup a_t is selected, its cooperative classification performance, such as accuracy, delay, and computation demand, are observed, and the algorithm updates its parameters for the subsequent time slots according to line 18 and 19 of the algorithm.

4.3 Summary

In this chapter, a subgroup scheduling optimization problem is formulated to determine the composition and role assignment of subgroups for cooperative classifications. The problem is reformulated as a MAB with cost subsidy problem after providing a brief introduction to the MAB theory and establishing a connection with the original problem. Algorithms which integrate a distance and line-of-sight based subgroup selection criteria with the cost-subsidized MAB theory are proposed to solve the problem in an online learning manner.

Chapter 5

Performance Evaluation

5.1 Simulation Setup

Simulations are conducted in a synthetic driving scenario involving a group of N AVs follow an object of interest in a three-lane, unidirectional road over 100 times slots. Each lane has a width of 3m. The object of interest is positioned at the front of the middle lane. The initial positions of each AV are randomly assigned on the multi-lane road with certain practical considerations. The X-coordinates of AVs are randomly chosen within a range of 3m to 100m from the object of interest, where the 3m boundary ensures that no AV is ahead of the object of interest, and 100m is observation range of camera [84]. The Y-coordinate for each AV is derived by locating the center of its randomly assigned lane and adding a slight random perturbation to introduce variability and realism to the positioning. The perturbation is generated from $\mathcal{N}(0, 0.2^2)$, ensuring that while vehicles are

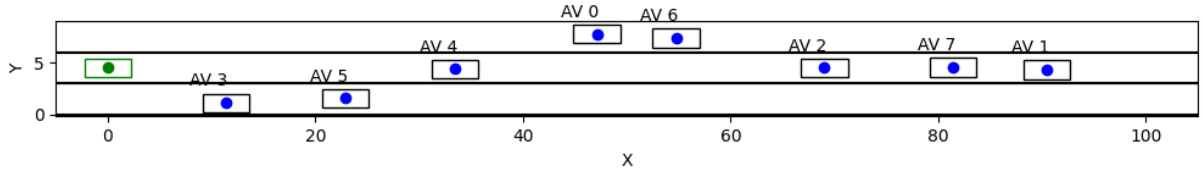


Figure 5.1: An example of the spatial arrangement of a group of 8 AVs (blue) and the object of interest (green) within the synthetic driving scenario.

generally centered in their lanes, there is a small degree of deviation to their exact position. In addition, both the AVs and the object of interest have dimensions of 1.8m in width and 4.5m in length and a minimum distance of 3m is imposed both between AVs and between an AV and the object of interest to avoid overlapping positions and ensure adequate spacing for driving safety. Figure 5.1 provides an example of the spatial arrangement of AVs and the object of interest within this synthetic driving scenario. We focus on a stationary setting under this scenario where the relative positions of AVs and the object of interest remain fixed throughout all time slots. For each AV $k \in \mathcal{K}$, its available CPU frequency $f_k(t)$ at each time slot is randomly sampled from a normal distribution. Recall that F_k represents the maximum CPU frequency of AV k, the mean of the distribution is uniformly distributed between $60\%F_k$ and $85\%F_k$, and the standard deviation of the distribution is uniformly distributed between $1\%F_k$ and $5\%F_k$. Such distribution captures the inherent characteristics of the computing resource availability for each AV over the simulation period. The wireless channel power gain is modeled by: $g_k(t) = A_0 d^{-2}$, with A_0 represents the path-loss coefficient and d corresponds to the distance between the helper and the aggregator [85]. Other simulation parameters are summarized in Table 5.1 [41, 72, 86].

The object of interest is represented by a 3D car model. We use images of dimension

Table 5.1: Simulation parameters.

Parameters	Value
Reward (Υ)	5
Bandwidth (B)	5 MHz
Received Noise power (σ^2)	1×10^{-13} W
Transmit power (p)	0.1 W
Path-loss coefficient (A_0)	-17.8 dB
Maximum computing frequency at AV k (F_k)	10 GHz
Classification accuracy requirement (Θ)	80 %
Classification delay requirement (τ)	350 ms
Energy efficiency coefficient (κ)	10^{-28}
Number of double-precision floating-point operations per cycle (ω)	8
Number of bits for a double-precision floating-point number (δ)	64

224×224, rendered from different perspectives of the model, to simulate the multi-view image data captured by AVs across different lanes. Specifically, for each AV, based on its lane, an image of the model, rendered from a lane-specific viewpoint, is used to mimic the different viewing perspectives of AVs. When the line of sight between an AV and the object of interest is blocked by another AV, an occlusion rectangle is applied to its image of the object of interest, concealing a portion of the model to simulate visual impairment. The occlusion rectangle has a base dimensions of 70×70. However, these dimensions can fluctuate, being multiplied by a factor ranging from 0.8 to 1.2, to ensure a degree of randomness. This randomness is introduced to better mimic real-world scenarios where the size and shape of the obstructions cannot be accurately determined. In addition, the image of each AV is further processed based on its proximity to the object of interest. If the distance between AV and the object of interest is closer than a predefined near distance, i.e., $D_{\text{near}} = 10$m, the image is cropped, zooming into the central region. The cropping

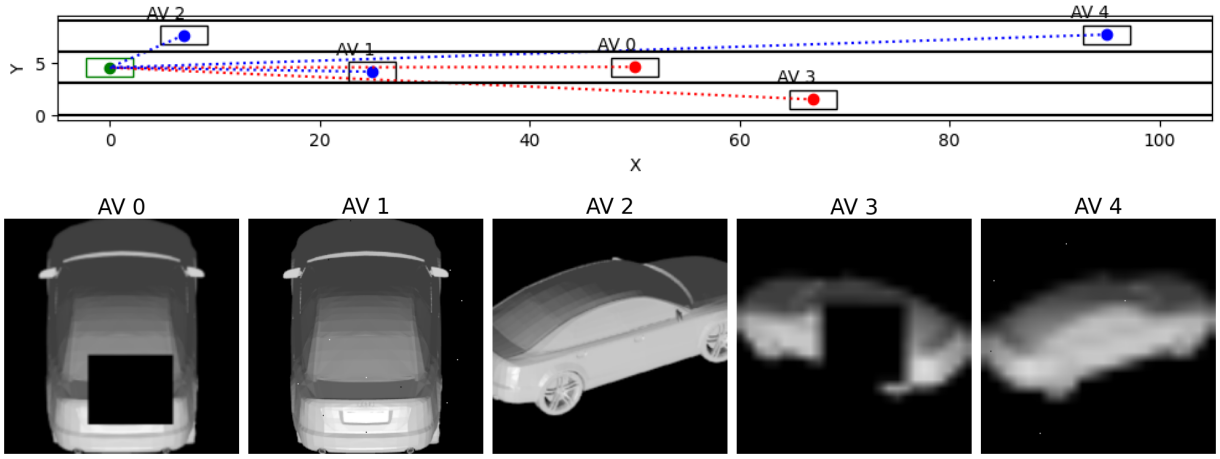


Figure 5.2: An example of how the images of the object of interest are processed for AVs under different viewing conditions. AVs with clear, unobstructed line of sight to the object of interest are depicted in blue, while AVs face obstructions in their line of sight are depicted in red.

scale is directly proportional to the distance between AV and the object, The closer the AV is to the object, the more the image is cropped towards its center. The cropped image is subsequently resized back to its original dimensions, simulating the effect that nearer objects often appear more truncated in images. Conversely, if the AV is farther away from the object than a predefined far distance, i.e., $D_{\text{far}} = 40\text{m}$, we reduced the size of the image by a factor of l_k, and increase it back using bilinear interpolation. The factor is proportional to distance score of the AV, and can be expressed as $l_k = 0.9S_k^{(D)} + 0.1$. The resulting image has reduced resolution and blurred image details. This technique is commonly used to simulate images taken for small or distant objects as these images are typically captured at a lower resolution [87]. Finally, noise is added to the images by randomly selecting 0.01% of the image pixels and subjecting them to salt and pepper noise. This simulates the transmission errors and sensor disturbances that arise due to dynamic

environmental factors. Figure 5.2 illustrates an example of how the images of the object of interest are processed for AVs under different viewing conditions.

Table 5.2: Feature Extraction and Classification Modules

Module	Layer	Input Dimension	Output Dimension	Filter Dimension
Feature Extraction	CONV1	(224, 224, 3)	(224, 224, 64)	(3, 3)
	RELU1	(224, 224, 64)	(224, 224, 64)	N/A
	POOL1	(224, 224, 64)	(112, 112, 64)	N/A
	CONV2	(112, 112, 64)	(112, 112, 128)	(3, 3)
	RELU2	(112, 112, 128)	(112, 112, 128)	N/A
	POOL2	(112, 112, 128)	(56, 56, 128)	N/A
	CONV3	(56, 56, 128)	(56, 56, 256)	(3, 3)
	RELU3	(56, 56, 256)	(56, 56, 256)	N/A
	CONV4	(56, 56, 256)	(56, 56, 256)	(3, 3)
	RELU4	(56, 56, 256)	(56, 56, 256)	N/A
	POOL3	(56, 56, 256)	(28, 28, 256)	N/A
	CONV5	(28, 28, 256)	(28, 28, 512)	(3, 3)
	RELU5	(28, 28, 512)	(28, 28, 512)	N/A
	CONV6	(28, 28, 512)	(28, 28, 512)	(3, 3)
	RELU6	(28, 28, 512)	(28, 28, 512)	N/A
	POOL4	(28, 28, 512)	(14, 14, 512)	N/A
	CONV7	(14, 14, 512)	(14, 14, 512)	(3, 3)
	RELU7	(14, 14, 512)	(14, 14, 512)	N/A
	CONV8	(14, 14, 512)	(14, 14, 512)	(3, 3)
	RELU8	(14, 14, 512)	(14, 14, 512)	N/A
	POOL5	(14, 14, 512)	(7, 7, 512)	N/A
Classification	FC1	7 * 7 * 512	4096	N/A
	RELU9	4096	4096	N/A
	FC2	4096	4096	N/A
	RELU10	4096	4096	N/A
	FC3	4096	40	N/A

A multi-view CNN model, adopting the VGG-11 architecture as the backbone network, is trained for cooperative classifications. The VGG-11 architecture is composed primarily of 11 weight layers, including eight `CONV` layers and three `FC` layers [88]. Small convolutional

filters of dimension 3×3 is used in each `CONV` layer. `MaxPool` layers interspersed between the `CONV` layers to reduce the spatial dimensions of the feature maps. `ReLu` activation layers are applied after each `CONV` and `FC` layer, with the exception of $\texttt{FC}_3$, which is followed by a `Softmax` classification layer to produce a probability distribution over all possible classes. We insert the `ViewPool` layer between $\texttt{Pool}_5$ and $\texttt{FC}_1$ to construct the three module architecture of multi-view classification model. The data for feature transmission corresponds to the output of feature extraction module, which is output data of $\texttt{Pool}_5$ layer. To account for the unpredictability of compression ratio, the data size of compressed feature data $\psi_k(t)$ for helper k at time slot t is set to be uniformly distributed from $70\%\psi$ to $100\%\psi$, where ψ represents the data size of uncompressed feature data. The specific layer configurations and layer parameters for feature extraction module and classification module are given in Table 5.2. The model is trained by employing the setup and multi-view 3D model dataset provided in [89]. The dataset contains training images of dimensions 224×224, generated by rendering models from the ModelNet40 dataset in 12 different views. Specifically, images are captured by cameras positioned at intervals of 30 degrees, directed towards the center of the object, with an elevation angle of 30 degrees. The resulting model is capable of aggregating information from a variable number of views, ranging from 1 to 12 views, without requiring a specific view ordering and outputs a single probability vector for multi-view image inputs of an object. During each time slot, each AV in the selected subgroup provides its image of the object of interest to the multi-view CNN model. AVs with different roles execute different modules of the model. The computation workload generated from executing these modules is computed based on their corresponding layer parameters and the equations provided in Section 3.2. The model generates a probability

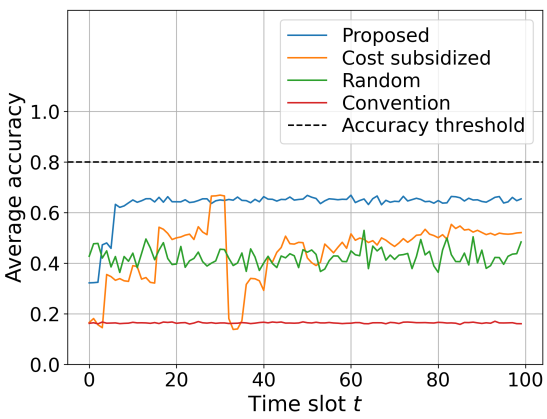
vector containing the predicted probability for all possible classes, from which the true class probability is determined. Such process is illustrated in Figure 1.4.

5.2 Simulation Results

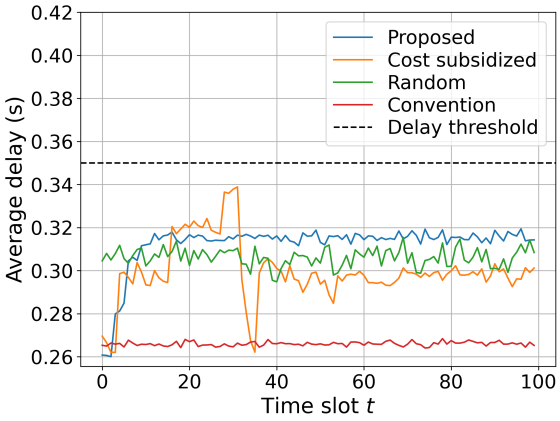
We simulate four distinct group sizes in the synthetic driving scenario, i.e., $N \in \{4, 6, 8, 10\}$, to assess the performance of various classification schemes and subgroup scheduling algorithms under different vehicle density. These include a group of 4 vehicles with 32 possible subgroup configurations, a 6-vehicle group with 192 possible subgroup configurations, an 8-vehicle group with 1024 possible subgroup configurations, and a 10-vehicle group that offers up to 5120 possible subgroup configurations. For every group size, 50 episodes are simulated, each containing a random vehicle position distribution and 100 time slots. At the beginning of each episode, the positions of AVs are randomly assigned as described in Section 5.1. However, the relative positions between the AVs and the object of interest remain static throughout the duration of the episode. During each time slot of each episode, we evaluate four distinct strategies. Among these, three employ cooperative classification scheme with different subgroup scheduling algorithms: the proposed algorithm, which is denoted in the figures as *Proposed*; the original cost subsidized MAB algorithm, denoted as *Cost subsidized*; and a random subgroup selection algorithm, denoted as *Random*. The fourth strategy is the conventional non-cooperative classification scheme, which is denoted in the figures as *Convention*. In the proposed subgroup scheduling algorithm, Algorithm 3 and Algorithm 4 are jointly utilized, with the exploration parameters β_1 set to 3 and β_2 set to 1 and the cost-subsidy factor ρ set to 0.1, enabling cost-subsidized MAB

algorithm to operate on a reduced subgroup exploration space. In the cost-subsidized subgroup scheduling algorithm, the unreduced subgroup exploration space is utilized by Algorithm 4 and the number of exploration for each subgroup is set to 1, i.e., $\beta_2 = 1$ and the cost-subsidy factor ρ is set to 0.1. The aim of the algorithm is to explore all possible subgroups to identify whether any particular subgroup is capable of providing tolerable classification performance, while requiring the least computation demand. On the other hand, in random subgroup scheduling algorithm, the selection of subgroup is done randomly in each time slot, without considering the specific cooperative classification performance of each subgroup. Lastly, we include the conventional non-cooperative scheme, where each AV in the group independently performs classification using local sensing data and computing resource to execute both feature extraction module and classification module throughout each episode, in our algorithm performance comparison as a benchmark scheme. For all strategies, the classification accuracies, delays, and computation demands are recorded in every time slot across all episodes.

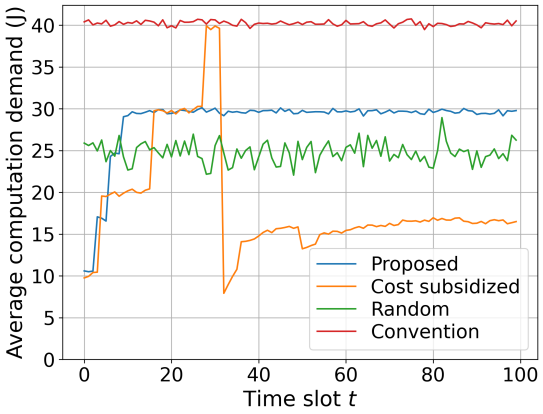
The performance of different strategies is evaluated across driving scenarios with different vehicle densities. Figure 5.3 showcases a comparison of average classification performance per time slot among different strategies for groups with only 4 AVs. The average classification performance for each time slot is obtained by averaging the values of classification accuracy, delay, and computation demand for that specific time slot over all 50 episodes. We observe that, for all four strategies, their average classification accuracy in each time slot falls below the accuracy threshold, while their average classification delay in each time slot remains within the acceptable delay threshold. The short delay is attributed to the small group size; even when cooperation is activated among all AVs in



(a) Average accuracy comparison



(b) Average delay comparison



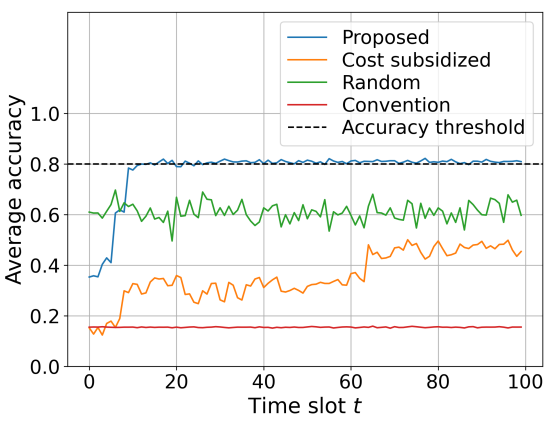
(c) Average computation demand comparison

Figure 5.3: Average classification performance per time slot for group of 4 AVs

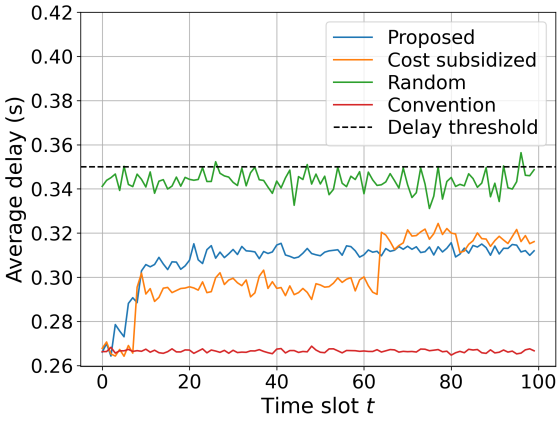
the group, the associated communication overhead remains relatively insignificant due to the small number of feature transmission. The underlying reason for poor classification accuracy performance is that, with a small group size, it is frequent for all vehicles in the group to be positioned in non-ideal observation locations, such as being positioned distant from the object of interest or all aligned within a single lane. In such scenarios, even with the adoption of cooperative classification scheme, the cooperative classification accuracy

remains subpar due to all AVs experiencing unfavorable viewing conditions relative to the object of interest. However, because of the exploration mechanism of Algorithm 3, the proposed algorithm progressively expands the size of the subgroup in an attempt to satisfy the classification requirement, ultimately involving all AVs in the group into the cooperative classification process. This explains why it exhibits superior classification accuracy performance compared to other strategies and require more computation demand than random subgroup scheduling algorithm and cost-subsidized MAB algorithm on average. On the other hand, while the cost-subsidized MAB algorithm does not attain the same classification accuracy level as the proposed algorithm, it necessitates the least computation demand among all strategies to achieve the classification accuracy performance that is closest to that of the proposed algorithm once its performance stabilizes. This is attributed to the fact that a group with a smaller size possesses a limited exploration space, i.e., small number of possible subgroup configurations. This limited exploration space enables the cost-subsidized MAB algorithm to complete its performance exploration of all subgroups, as depicted by the stair-like patterns in the subfigures, and identify the subgroup that requires lowest cost, i.e., computation demand, to achieve a tolerable level of QoS before an episode concludes. However, as shown in all subfigures, such exhaustive exploration leads to a significantly longer convergence time when compared to the proposed algorithm.

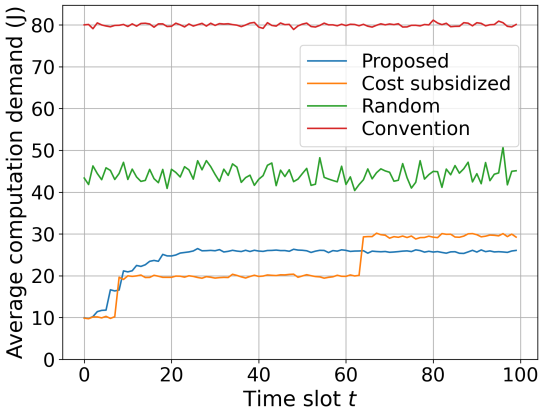
Figure 5.4 presents a similar comparison of average classification performance per time slot across various strategies, but for groups comprising 8 AVs. The notable difference between these two comparison is that, for group with more AVs, the proposed algorithm begins to excel in every aspect of the classification performance. After a brief exploration phase, the proposed algorithm swiftly converges, achieving classification accuracy and in-



(a) Average accuracy comparison



(b) Average delay comparison



(c) Average computation demand comparison

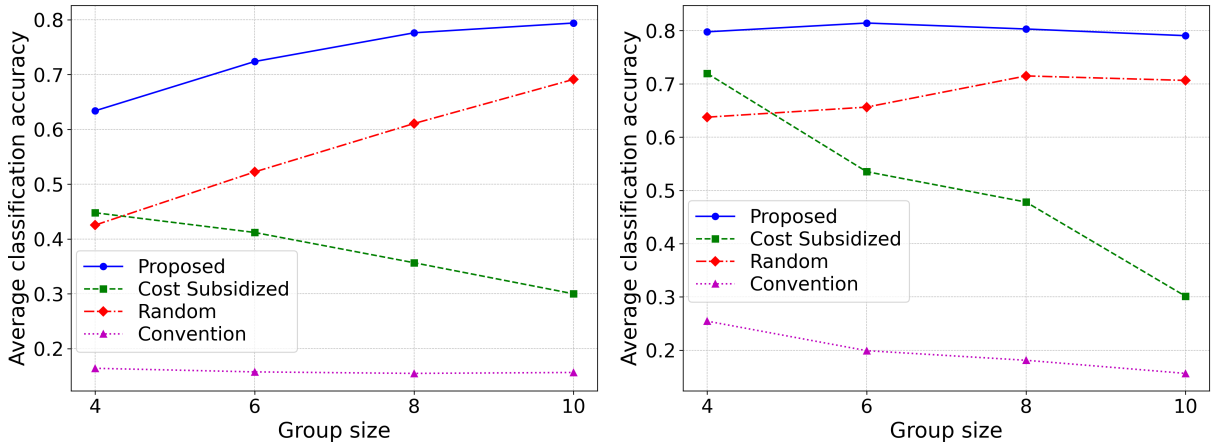
Figure 5.4: Average classification performance per time slot for group of 8 AVs

curring classification delay that consistently fall within the accuracy and delay thresholds. Additionally, in terms of computation demand, the proposed algorithm consumes the least energy for executing the cooperative classification task compared to all other strategies. The improvement in the performance of the proposed algorithm, as it transitions from a smaller to a larger group size, stems from the increased probability of having more vehicles within the optimal observation range and the vehicles being more dispersed as the group

size expands. As a result, the potential of the cooperative classification scheme can be exerted to its maximum extent through the selection of a few AVs possess superior viewing conditions for cooperative classification, as suggested in Algorithm 3. On the other hand, the performance of the cost-subsidized MAB algorithm deteriorates compared to its previous results when group size is 4. This decline is due to the substantial expansion of the subgroup exploration space, reflected by the significant increase in the number of subgroup configurations for each possible subgroup size. Such increase is shown by the elongated stair patterns in the figures. As a result, the algorithm is no longer able to complete its exploration of all subgroups and fails to converge within each episode. This leads to greater computation demand, reflected in higher computing energy, along with increased classification delay; yet, the algorithm still yields average classification accuracy per time slot inferior to that of the proposed algorithm.

We further showcase the average classification accuracy, delay, and normalized total computation demand of different strategies for all four group sizes. We present a comparative analysis of these performance metrics under two slightly different simulation settings. The first simulation setting is identical to the setting described in Section 5.1. Whereas for the second simulation setting, instead of assigning random positions for vehicles in the group, we ensures that a minimum of one vehicle in the group is located within the optimal observation range, i.e., $[D_{\text{near}}, D_{\text{far}}]$.

The average classification accuracy and delay obtained under each different group size are computed by averaging accuracy values and delay values over all time slots and episodes. As shown in Figure 5.5a, when AVs in the group are positioned randomly relative to the object of interest, the proposed algorithm exhibits enhanced classification accuracy as group



(a) Random vehicle positions

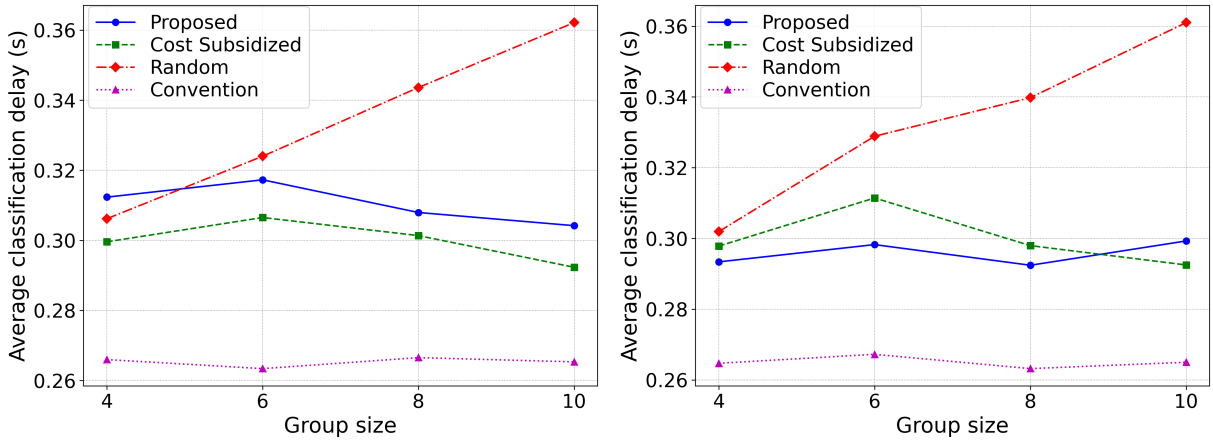
(b) Adjusted vehicle positions

Figure 5.5: Comparison of average classification accuracy across strategies and group sizes in two simulation settings

size expands. In contrast, the classification accuracy of the cost-subsidized MAB algorithm declines with an increasing group size. This trend is consistent with our prior observations and can be explained in a similar logic. The classification accuracy of the random subgroup scheduling algorithm also rises with the group size. The improvement of accuracy can be attributed to the tendency that more subgroups with larger sizes will be randomly selected as the group size expands. This tendency increases the probability of AVs with superior viewing conditions being included into selected subgroups, thereby increasing the cooperative classification accuracy. By intentionally positioning at least one AV in the group within the ideal observation range, an improvement in the classification accuracy performance can be observed for the proposed algorithm and the random subgroup scheduling algorithm, as shown in Figure 5.5b, especially when the group size is small. In addition, when compared the results achieved by random vehicle placement, there is also an improvement in the

average classification accuracy observed for cost-subsidized MAB algorithm and conventional non-cooperative scheme when the group size is 4. The improvement in classification accuracy for the four strategies can be attributed to the presence of more vehicles with favorable viewing conditions within the group, resulting from the adjusted vehicle positions. Thus, with improvements in viewing conditions at the individual vehicle level, it is logical to expect a corresponding rise in the performance of cooperative classification accuracy at the subgroup level, especially for algorithms that leverages vehicles viewing conditions as a metric for subgroup scheduling, i.e., the proposed algorithm. Nonetheless, in contrast to the proposed algorithm, which exhibits the most pronounced gains from the vehicle position adjustment and consistently achieves an average classification accuracy surpassing the accuracy threshold, the classification accuracy performance of the cost-subsidized MAB algorithm and the conventional non-cooperative scheme exhibit a decline as the group size increases. The decrease in accuracy for cost-subsidized MAB algorithm is attributed to the considerable expansion of subgroup exploration space along with the growth in group size. Similarly, the decrease in average accuracy for the conventional non-cooperative scheme is attributed to the increased number of presences of vehicles with inferior viewing conditions as the group size expands. As for random subgroup scheduling algorithm, despite its noticeable gain in classification accuracy through vehicle position adjustment, its overall accuracy performance remains inferior compared to the proposed algorithm, primarily due to its inherent random nature.

Figure 5.6 showcases the comparison of average classification delay for the previous two simulation settings. In both settings, the average classification delay incurred by the proposed algorithm and the cost-subsidized MAB algorithm consistently fall below



(a) Random vehicle positions

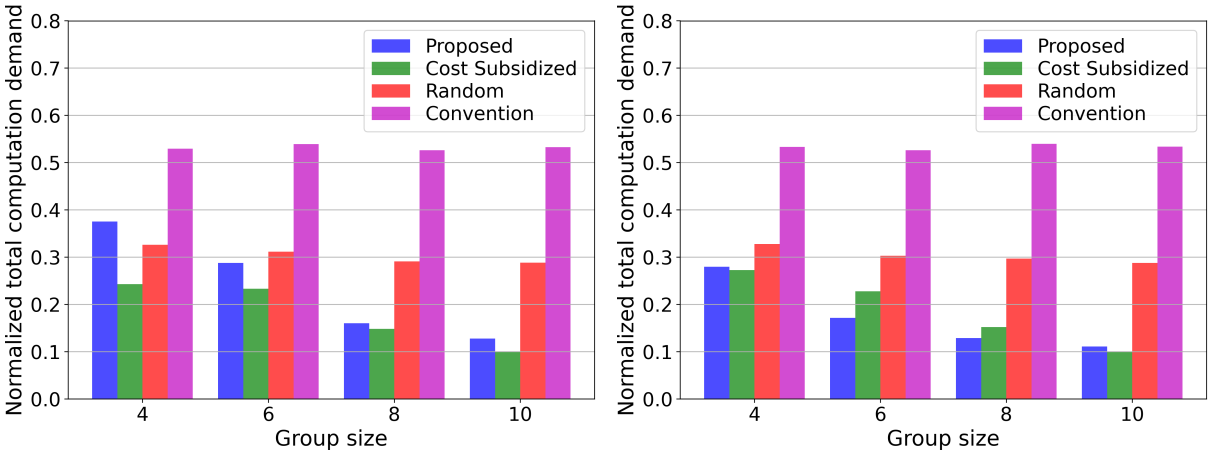
(b) Adjusted vehicle positions

Figure 5.6: Comparison of average classification delay across strategies and group sizes in two simulation settings

the maximum delay threshold for all group sizes. The containment of classification delay within acceptable range by the proposed algorithm is attributed to its mechanism of prioritizing smaller-sized subgroups, as suggested in Algorithm 3. Through a comparison between Figure 5.6a and Figure 5.6b, a noticeable reduction in overall classification delay performance across all group sizes can be observed for the proposed algorithm. Such reduction stems from the ability of the proposed algorithm to utilize fewer AVs for cooperative classification when a greater number of vehicles possess superior viewing conditions. On the other hand, the reason behind the cost-subsidized MAB algorithm achieving satisfactory classification delay lies in its inability to finish pure exploration phase within a limited period of time. This limitation becomes particularly pronounced when the algorithm is applied to a large group of vehicles, as shown in Figure 5.4, wherein the algorithm cannot even complete the exploration of the considerable number of configurations for small-sized

subgroups within 100 time slots. Consequently, the algorithm consistently experiences small cooperative classification delays. In addition, we observe that the classification delay incurred by random subgroup scheduling algorithm increase alongside the expansion of group size in both settings. This outcome is in line with our expectations, as an increase in group size means that more subgroups of large sizes will be randomly selected, necessitating additional time for feature transmission and consequently extending the classification delay. Beyond mere numerical evaluation, practical implementation of the cooperative classification scheme in vehicular networks could potentially lead to reduced cooperative classification delays. In our simulation, the bandwidth for feature transmission is set to 5 MHz. However, according to European Telecommunications Standards Institute (ETSI), the actual bandwidth allocated for vehicular communications in vehicular networks could surpass our simulated value [90]. This potential increase in bandwidth could further reduce delay incurred from feature transmission, enhancing the real-world feasibility of cooperative classification scheme.

The comparison of normalized total computation demand between different strategies and group sizes in the two simulation settings is shown in Figure 5.7. The normalized total computation demand is computed by first averaging the total computation demand in each episode over all episodes. The averaged value is then normalized with respect to the maximum total computation demand, which is derived by multiplying the maximum computation demand of a single vehicle by the number of time slots in an episode and the size of the group; the maximum computation demand for each vehicle indicates the computing energy consumption of an AV when it operates at its peak CPU frequency, executing both the feature extraction and classification modules. As observed by comparing Figure



(a) Random vehicle positions

(b) Adjusted vehicle positions

Figure 5.7: Comparison of total computation demand across strategies and group sizes in two simulation settings

5.7a and Figure 5.7b, the total computation demand required by the proposed algorithm has undergone reduction, especially for small group size, when at least one vehicle in the group is positioned within the optimal observation range. This phenomenon is expected due to the prevalence of vehicles in non-ideal observation positions in situations where the group size is small, and the distribution of vehicle positions are relatively more random. As a result, the proposed algorithm involves all vehicles in the cooperative classification process to enhance accuracy, leading to a relatively substantial computation demand for small group sizes. However, as the group size increases or as more vehicles are positioned within the optimal observation range, the proposed algorithm become capable of utilizing fewer vehicles with superior viewing conditions for cooperative classification, which allows it to perform better in terms of achieving better classification accuracy with less computation demand. In addition, the normalized total computation demand of the proposed

algorithm decrease as the group size increase in both settings. This trend signifies that the reduction in computation demand achieved by the proposed algorithm is positively related to the group size, with an increased number of vehicles in the group, the reduction in total computation demand becomes more pronounced. The cost-subsidized MAB algorithm exhibits similar relationship between the normalized total computation demand and group size. Nevertheless, its reduced computation demand stems from the partial exploration of subgroups with small sizes when the group size is large, and does not lead to an actual improvement in classification performance due to the concurrent decline in classification accuracy as group size increases.

To summarize, replacing non-cooperative classification scheme with cooperative classification scheme in object tracking process introduces both benefits and costs. While the cooperative classification scheme improves classification accuracy and reduces computation demand for executing classifications, this comes at the expense of wireless transmission resource consumption and increased classification delay, due to V2V communications. Moreover, the cooperative classification scheme also incurs signaling overhead, as coordinating the transmission of feature data from various vehicle sources becomes necessary for this scheme. However, these costs are justifiable when considering the classification performance improvements that cooperative classification scheme can provide.

5.3 Summary

In this chapter, the performance of the proposed cooperative classification scheme and the subgroup scheduling algorithm is numerically evaluated and compared against alter-

native strategies. We first evaluate the impact of group size on the efficacy of various strategies, yielding detailed insights into each. The efficacy of the proposed scheme and algorithm is validated through classification performance comparisons between different strategies and group sizes in two different simulation settings. The results and our observations highlight that the proposed scheme and algorithm are capable of accomplishing classifications with minimal computation demand, delivering enhanced classification accuracy and reduced delays relative to other strategies.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

This research aims to address the issues related to the susceptibility of classification accuracy to viewing conditions of vehicles and excessive computation demand in the object tracking process of autonomous driving. A cooperative classification scheme, which involve distributing the computation workload of classification among multiple AVs and enabling the sharing and aggregation of multi-view sensing data among those AVs, is investigated to improve the accuracy and computing efficiency of object classification tasks. To optimize the performance of the cooperative classification scheme, a subgroup scheduling problem, which aims to determine the optimal selection of AV subgroups for each cooperative classification over a given period of time, is studied under the challenges of lacking information regarding the cooperative classification performance of each subgroup. An

optimization problem is formulated to determine the composition and role assignment of each scheduled subgroup over the considered time period, with the objective of minimizing the total computation demand required by the group for cooperative classifications, while ensuring the satisfaction of classification accuracy and delay requirements. Two algorithms are proposed to solve the subgroup scheduling optimization problem. Drawing upon the insights concerning the selection of subgroups that best align with our optimization objective, the first algorithm enables a coarse-grained exploration of subgroup candidates to narrow down potential subgroup selections. The second algorithm utilizes developed tools from cost-subsidized MAB theory to perform fine-grained exploration on the resulting subgroups from the first algorithm, from which the optimal subgroups for cooperative classifications are determined. The simulation results validate the effectiveness of the proposed scheme and subgroup scheduling algorithm by showcasing their capability to provide superior classification accuracy and reduced classification delay with low computation demand, outperforming other schemes and algorithms in achieving superior QoS while maintaining high computing efficiency.

6.2 Future Research Work

Several future research directions are open to explore to improve the practicality and applicability of subgroup scheduling algorithm in real-world driving scenarios and promote the adoption of cooperative classification scheme:

- In Section 5.2, we demonstrated that by intentionally positioning at least one vehicle

within the optimal observation range, the proposed subgroup scheduling algorithm exhibits superior performance, even with small group sizes. This implies that the efficiency of the cooperative classification scheme is influenced by the vehicles' relative distance to the object of interest. Hence, decisions regarding cooperation, such as when a group of vehicles should utilize the cooperative classification scheme, must be judiciously made, especially with limited vehicles in the group. However, this study centers on subgroup scheduling for enhancing cooperative classification performance, assuming cooperation decisions are predetermined. Future research could delve into making cooperation decisions, exploring when vehicles should engage in collaboration, and when to employ the proposed cooperative classification scheme and the subgroup scheduling algorithm.

- This research focuses on a static setting, wherein the number of AVs in the considered group and relative positions of AVs in relation of the object of interest, remain constant throughout the considered time period. However, the group size can be subject to change due to the arrival or departure of AVs, leading to variations in the number of candidate subgroups available for each cooperative classification. Furthermore, as AVs navigate through different traffic scenarios and engage in actions, such as lane-changing and overtaking, factors such as occlusion relationships and illumination conditions may vary significantly, which could fundamentally affect the inherent cooperative classification performance for each subgroup and impact the subgroup scheduling decision. Therefore, a dynamic setting should be investigated in the future to account for the influence of arrivals and departures of AV and the ever-changing environmental conditions.

- The investigation in this research is limited to a single group of AVs and a single object of interest. This allows the group to fully utilize all available communication resources during the cooperative classification process, without the need for communication resource management. In future research, the investigation can be extended to include multiple groups of AVs and multiple objects of interest. The cooperative classification scheme and subgroup scheduling algorithm can be leveraged to enable a larger number of AVs to benefit from this approach. However, such extension introduces the challenges of efficiently managing the communication resources for each group, ensuring timely completion of cooperative classification by multiple subgroups, and maximizing the benefits received from employing cooperative classification scheme.

- In practical implementation of the cooperative classification scheme, individual vehicles might lack the memory space to store multi-view feature data for aggregation, or the computational power to swiftly execute cooperative classifications. Infrastructures, such as edge servers, can be incorporated into the existing system to enhance memory capacity and overall computing capability of vehicle group, cooperation and joint resource management between AVs and edge servers can then be investigated to further improve the delay performance of cooperative classification.

References

[1] Sae International. Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles. *SAE international*, 4970(724):1–5, 2018.

[2] Hongbo Gao, Bo Cheng, Jianqiang Wang, Keqiang Li, Jianhui Zhao, and Deyi Li. Object classification using CNN-based fusion of vision and lidar in autonomous vehicle environment. *IEEE Transactions on Industrial Informatics*, 14(9):4224–4231, 2018.

[3] Eduardo Arnold, Omar Y Al-Jarrah, Mehrdad Dianati, Saber Fallah, David Oxtoby, and Alex Mouzakitis. A survey on 3D object detection methods for autonomous driving applications. *IEEE Transactions on Intelligent Transportation Systems*, 20(10):3782–3795, 2019.

[4] Shengping Zhang, Yuankai Qi, Feng Jiang, Xiangyuan Lan, Pong C Yuen, and Huiyu Zhou. Point-to-set distance metric learning on deep representations for visual tracking. *IEEE Transactions on Intelligent Transportation Systems*, 19(1):187–198, 2017.

[5] Sajjad Mozaffari, Omar Y Al-Jarrah, Mehrdad Dianati, Paul Jennings, and Alexandros Mouzakitis. Deep learning-based vehicle behavior prediction for autonomous

driving applications: A review. *IEEE Transactions on Intelligent Transportation Systems*, 23(1):33–47, 2020.

[6] Kichang Yang, Juheon Yi, Kyungjin Lee, and Youngki Lee. Flexpatch: Fast and accurate object detection for on-device high-resolution live video analytics. In *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*, pages 1898–1907, 2022.

[7] Jianbing Shen, Xin Tang, Xingping Dong, and Ling Shao. Visual object tracking by hierarchical attention siamese network. *IEEE transactions on cybernetics*, 50(7):3068–3080, 2019.

[8] Yiming Li, Changhong Fu, Ziyuan Huang, Yinqiang Zhang, and Jia Pan. Intermittent contextual learning for keyfilter-aware UAV object tracking using deep convolutional feature. *IEEE Transactions on Multimedia*, 23:810–822, 2020.

[9] Miaomiao Liu, Xianzhong Ding, and Wan Du. Continuous, real-time object detection on mobile devices without offloading. In *2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS)*, pages 976–986, 2020.

[10] Bo Yang, Xuelin Cao, Chau Yuen, and Lijun Qian. Offloading optimization in edge computing for deep-learning-enabled target tracking by Internet of UAVs. *IEEE Internet of Things Journal*, 8(12):9878–9893, 2020.

[11] Di Feng, Ali Harakeh, Steven L Waslander, and Klaus Dietmayer. A review and comparative study on probabilistic object detection in autonomous driving. *IEEE Transactions on Intelligent Transportation Systems*, 23(8):9961–9980, 2021.

[12] Di Feng, Christian Haase-Schütz, Lars Rosenbaum, Heinz Hertlein, Claudius Glaeser, Fabian Timm, Werner Wiesbeck, and Klaus Dietmayer. Deep multi-modal object detection and semantic segmentation for autonomous driving: Datasets, methods, and challenges. *IEEE Transactions on Intelligent Transportation Systems*, 22(3):1341–1360, 2020.

[13] Ying Li, Lingfei Ma, Zilong Zhong, Fei Liu, Michael A Chapman, Dongpu Cao, and Jonathan Li. Deep learning for lidar point clouds in autonomous driving: A review. *IEEE Transactions on Neural Networks and Learning Systems*, 32(8):3412–3432, 2020.

[14] Xumiao Zhang, Anlan Zhang, Jiachen Sun, Xiao Zhu, Y Ethan Guo, Feng Qian, and Z Morley Mao. Emp: Edge-assisted multi-vehicle perception. In *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking*, pages 545–558, 2021.

[15] Yukuan Jia, Ruiqing Mao, Yuxuan Sun, Sheng Zhou, and Zhisheng Niu. Online V2X scheduling for raw-level cooperative perception. In *ICC 2022-IEEE International Conference on Communications*, pages 309–314, 2022.

[16] Qi Chen, Sihai Tang, Qing Yang, and Song Fu. Cooper: Cooperative perception for connected autonomous vehicles based on 3D point clouds. In *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, pages 514–524, 2019.

[17] Alaa Awad Abdellatif, Carla Fabiana Chiasserini, Francesco Malandrino, Amr Mohamed, and Aiman Erbad. Active learning with noisy labelers for improving classifi-

cation accuracy of connected vehicles. *IEEE Transactions on Vehicular Technology*, 70(4):3059–3070, 2021.

[18] 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 Journal of Vehicular Technology*, 2:78–93, 2021.

[19] 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 Transactions on Intelligent Transportation Systems*, 16(2):663–680, 2014.

[20] 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 *Proceedings of the 4th ACM/IEEE Symposium on Edge Computing*, pages 88–100, 2019.

[21] Qing Yang, Song Fu, Honggang Wang, and Hua Fang. Machine-learning-enabled cooperative perception for connected autonomous vehicles: Challenges and opportunities. *IEEE Network*, 35(3):96–101, 2021.

[22] Mohamed K Abdel-Aziz, Cristina Perfecto, Sumudu Samarakoon, Mehdi Bennis, and Walid Saad. Vehicular cooperative perception through action branching and federated reinforcement learning. *IEEE Transactions on Communications*, 70(2):891–903, 2021.

[23] Boubakr Nour, Spyridon Mastorakis, and Abderrahmen Mtibaa. Compute-less networking: Perspectives, challenges, and opportunities. *IEEE network*, 34(6):259–265, 2020.

[24] Jonathan Lee, Abderrahmen Mtibaa, and Spyridon Mastorakis. A case for compute reuse in future edge systems: An empirical study. In *2019 IEEE Globecom Workshops (GC Wkshps)*, pages 1–6, 2019.

[25] Dongyun Lin, Yiqun Li, Yi Cheng, Shitala Prasad, Aiyuan Guo, and Yanpeng Cao. Multi-range view aggregation network with vision transformer feature fusion for 3D object retrieval. *IEEE Transactions on Multimedia*, 2023.

[26] Yong Xu, Chaoda Zheng, Ruotao Xu, Yuhui Quan, and Haibin Ling. Multi-view 3D shape recognition via correspondence-aware deep learning. *IEEE Transactions on Image Processing*, 30:5299–5312, 2021.

[27] Kai Sun, Jiangshe Zhang, Junmin Liu, Ruixuan Yu, and Zengjie Song. DRCNN: Dynamic routing convolutional neural network for multi-view 3D object recognition. *IEEE Transactions on Image Processing*, 30:868–877, 2020.

[28] Kui Jia, Jiehong Lin, Mingkui Tan, and Dacheng Tao. Deep multi-view learning using neuron-wise correlation-maximizing regularizers. *IEEE Transactions on Image Processing*, 28(10):5121–5134, 2019.

[29] Xinwei He, Song Bai, Jiajia Chu, and Xiang Bai. An improved multi-view convolutional neural network for 3D object retrieval. *IEEE Transactions on Image Processing*, 29:7917–7930, 2020.

[30] Ze Yang and Liwei Wang. Learning relationships for multi-view 3D object recognition. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 7505–7514, 2019.

[31] Yifan Feng, Zizhao Zhang, Xibin Zhao, Rongrong Ji, and Yue Gao. GVCNN: Group-view convolutional neural networks for 3D shape recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 264–272, 2018.

[32] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view convolutional neural networks for 3D shape recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 945–953, 2015.

[33] Surat Teerapittayanon, Bradley McDanel, and Hsiang-Tsung Kung. Distributed deep neural networks over the cloud, the edge and end devices. In *2017 IEEE 37th international conference on distributed computing systems (ICDCS)*, pages 328–339. IEEE, 2017.

[34] Jinhang Choi, Zeinab Hakimi, John Sampson, and Vijaykrishnan Narayanan. Byzantine-tolerant inference in distributed deep intelligent system: Challenges and opportunities. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 9(3):509–519, 2019.

[35] Rasheed Hussain, Jooyoung Lee, and Sherali Zeadally. Trust in VANET: A survey of current solutions and future research opportunities. *IEEE transactions on intelligent transportation systems*, 22(5):2553–2571, 2020.

[36] Zhong-Qiu Zhao, Peng Zheng, Shou-tao Xu, and Xindong Wu. Object detection with deep learning: A review. *IEEE transactions on neural networks and learning systems*, 30(11):3212–3232, 2019.

[37] Jian-Xun Mi, An-Di Li, and Li-Fang Zhou. Review study of interpretation methods for future interpretable machine learning. *IEEE Access*, 8:191969–191985, 2020.

[38] Xiaoshuang Shi, Fuyong Xing, Kaidi Xu, Pingjun Chen, Yun Liang, Zhiyong Lu, and Zhenhua Guo. Loss-based attention for interpreting image-level prediction of convolutional neural networks. *IEEE Transactions on Image Processing*, 30:1662–1675, 2020.

[39] Fred Hohman, Haekyu Park, Caleb Robinson, and Duen Horng Polo Chau. Summit: Scaling deep learning interpretability by visualizing activation and attribution summarizations. *IEEE transactions on visualization and computer graphics*, 26(1):1096–1106, 2019.

[40] Zhuwei Qin, Fuxun Yu, Chenchen Liu, and Xiang Chen. How convolutional neural network see the world: A survey of convolutional neural network visualization methods. *arXiv preprint arXiv:1804.11191*, 2018.

[41] 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 Transactions on vehicular technology*, 68(4):3061–3074, 2019.

[42] Jiawei Shao, Yuyi Mao, and Jun Zhang. Task-oriented communication for multi-device cooperative edge inference. *IEEE Transactions on Wireless Communications*, 2022.

[43] Zhiwei Hao, Guanyu Xu, Yong Luo, Han Hu, Jianping An, and Shiwen Mao. Multi-agent collaborative inference via DNN decoupling: Intermediate feature compression and edge learning. *IEEE Transactions on Mobile Computing*, 2022.

[44] Jingyun Feng, Zhi Liu, Celimuge Wu, and Yusheng Ji. AVE: Autonomous vehicular edge computing framework with ACO-based scheduling. *IEEE Transactions on Vehicular Technology*, 66(12):10660–10675, 2017.

[45] Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multi-armed bandit problem. *Machine learning*, 47:235–256, 2002.

[46] Sébastien Bubeck, Nicolo Cesa-Bianchi, et al. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends in Machine Learning*, 5(1):1–122, 2012.

[47] Yuxuan Sun, Sheng Zhou, and Jie Xu. EMM: Energy-aware mobility management for mobile edge computing in ultra dense networks. *IEEE Journal on Selected Areas in Communications*, 35(11):2637–2646, 2017.

[48] Antoine Caillot, Safa Ouerghi, Pascal Vasseur, Rémi Boutteau, and Yohan Dupuis. Survey on cooperative perception in an automotive context. *IEEE Transactions on Intelligent Transportation Systems*, 23(9):14204–14223, 2022.

[49] Azim Eskandarian, Chaoxian Wu, and Chuanyang Sun. Research advances and challenges of autonomous and connected ground vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 22(2):683–711, 2019.

[50] Eduardo Arnold, Mehrdad Dianati, Robert de Temple, and Saber Fallah. Cooperative perception for 3D object detection in driving scenarios using infrastructure sensors. *IEEE Transactions on Intelligent Transportation Systems*, 23(3):1852–1864, 2020.

[51] Zongdian Li, Tao Yu, Ryuichi Fukatsu, Gia Khanh Tran, and Kei Sakaguchi. Proof-of-concept of a SDN based mmwave V2X network for safe automated driving. In *2019 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6. IEEE, 2019.

[52] Junil Choi, Vutha Va, Nuria Gonzalez-Prelcic, Robert Daniels, Chandra R Bhat, and Robert W Heath. Millimeter-wave vehicular communication to support massive automotive sensing. *IEEE Communications Magazine*, 54(12):160–167, 2016.

[53] Seong-Woo 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 Intelligent Transportation Systems Magazine*, 7(3):39–50, 2015.

[54] Tsun-Hsuan Wang, Sivabalan Manivasagam, Ming Liang, Bin Yang, Wenyuan Zeng, and Raquel Urtasun. V2vnet: Vehicle-to-vehicle communication for joint perception and prediction. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 605–621. Springer, 2020.

[55] Shaohua Qi, Xin Ning, Guowei Yang, Liping Zhang, Peng Long, Weiwei Cai, and Weijun Li. Review of multi-view 3D object recognition methods based on deep learning. *Displays*, 69:102053, 2021.

[56] Charles R Qi, Hao Su, Matthias Nießner, Angela Dai, Mengyuan Yan, and Leonidas J Guibas. Volumetric and multi-view CNNs for object classification on 3D data. In

Proceedings of the IEEE conference on computer vision and pattern recognition, pages 5648–5656, 2016.

[57] Ian Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron Courville, and Yoshua Bengio. Maxout networks. In *International conference on machine learning*, pages 1319–1327. PMLR, 2013.

[58] Eduardo Arnold, Omar Y Al-Jarrah, Mehrdad Dianati, Saber Fallah, David Oxtoby, and Alex Mouzakitis. Cooperative object classification for driving applications. In *2019 IEEE Intelligent Vehicles Symposium (IV)*, pages 2484–2489, 2019.

[59] Chu Wang, Marcello Pelillo, and Kaleem Siddiqi. Dominant set clustering and pooling for multi-view 3D object recognition. *arXiv preprint arXiv:1906.01592*, 2019.

[60] Xiaobo Zhao, Minoo Hosseinzadeh, Nathaniel Hudson, Hana Khamfroush, and Daniel E Lucani. Improving the accuracy-latency trade-off of edge-cloud computation offloading for deep learning services. In *2020 IEEE Globecom Workshops (GC Wkshps*, pages 1–6. IEEE, 2020.

[61] Shangbin Han, Qianhong Wu, Han Zhang, Bo Qin, Jiankun Hu, Xingang Shi, Linfeng Liu, and Xia Yin. Log-based anomaly detection with robust feature extraction and online learning. *IEEE Transactions on Information Forensics and Security*, 16:2300–2311, 2021.

[62] Bo Yang, Xuelin Cao, Xiangfang Li, Qinqing Zhang, and Lijun Qian. Mobile-edge-computing-based hierarchical machine learning tasks distribution for iiot. *IEEE Internet of Things Journal*, 7(3):2169–2180, 2019.

[63] Changsheng You, Kaibin Huang, Hyukjin Chae, and Byoung-Hoon Kim. Energy-efficient resource allocation for mobile-edge computation offloading. *IEEE Transactions on Wireless Communications*, 16(3):1397–1411, 2016.

[64] Mahdi Ben Ghorbel, David Rodriguez-Duarte, Hakim Ghazzai, Md Jahangir Hossain, and Hamid Menouar. Joint position and travel path optimization for energy efficient wireless data gathering using unmanned aerial vehicles. *IEEE Transactions on Vehicular Technology*, 68(3):2165–2175, 2019.

[65] Bishmita Hazarika, Keshav Singh, Sudip Biswas, and Chih-Peng Li. DRL-based resource allocation for computation offloading in IoV networks. *IEEE Transactions on Industrial Informatics*, 18(11):8027–8038, 2022.

[66] Yuxuan Sun, Jinhui Song, Sheng Zhou, Xueying Guo, and Zhisheng Niu. Task replication for vehicular edge computing: A combinatorial multi-armed bandit based approach. In *2018 IEEE Global Communications Conference (GLOBECOM)*, pages 1–7. IEEE, 2018.

[67] Yukuan Jia, Ruiqing Mao, Yuxuan Sun, Sheng Zhou, and Zhisheng Niu. MASS: Mobility-aware sensor scheduling of cooperative perception for connected automated driving. *arXiv preprint arXiv:2302.13029*, 2023.

[68] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[69] Ramyad Hadidi, Jiashen Cao, Michael S Ryoo, and Hyesoon Kim. Toward collaborative inferencing of deep neural networks on Internet-of-Things devices. *IEEE Internet of Things Journal*, 7(6):4950–4960, 2020.

[70] Young Geun Kim and Carole-Jean Wu. Autoscale: Energy efficiency optimization for stochastic edge inference using reinforcement learning. In *2020 53rd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, pages 1082–1096. IEEE, 2020.

[71] Yundong Zhang, Naveen Suda, Liangzhen Lai, and Vikas Chandra. Hello edge: Keyword spotting on microcontrollers. *arXiv preprint arXiv:1711.07128*, 2017.

[72] Kaige Qu, Weihua Zhuang, Wen Wu, Mushu Li, Xuemin Shen, Xu Li, and Weisen Shi. Stochastic cumulative dnn inference with rl-aided adaptive iot device-edge collaboration. *IEEE Internet of Things Journal*, 2023.

[73] Zehong Lin, Suzhi Bi, and Ying-Jun Angela Zhang. Optimizing ai service placement and resource allocation in mobile edge intelligence systems. *IEEE Transactions on Wireless Communications*, 20(11):7257–7271, 2021.

[74] Giuseppe Burtini, Jason Loeppky, and Ramon Lawrence. A survey of online experiment design with the stochastic multi-armed bandit. *arXiv preprint arXiv:1510.00757*, 2015.

[75] Deeksha Sinha, Karthik Abinav Sankararaman, Abbas Kazerouni, and Vashist Avadhanula. Multi-armed bandits with cost subsidy. In *International Conference on Artificial Intelligence and Statistics*, pages 3016–3024. PMLR, 2021.

[76] Tingting Yang, Shan Gao, Jiabo Li, Meng Qin, Xin Sun, Ran Zhang, Miao Wang, and Xianbin Li. Multi-armed bandits learning for task offloading in maritime edge intelligence networks. *IEEE Transactions on Vehicular Technology*, 71(4):4212–4224, 2022.

[77] Xinzhu Ma, Yinmin Zhang, Dan Xu, Dongzhan Zhou, Shuai Yi, Haojie Li, and Wanli Ouyang. Delving into localization errors for monocular 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4721–4730, 2021.

[78] Christian Eggert, Stephan Brehm, Anton Winschel, Dan Zecha, and Rainer Lienhart. A closer look: Small object detection in faster r-cnn. In *2017 IEEE international conference on multimedia and expo (ICME)*, pages 421–426. IEEE, 2017.

[79] Michał Koziarski and Bogusław Cyganek. Impact of low resolution on image recognition with deep neural networks: An experimental study. *International Journal of Applied Mathematics and Computer Science*, 28(4):735–744, 2018.

[80] Shane Gilroy, Edward Jones, and Martin Glavin. Overcoming occlusion in the automotive environment—a review. *IEEE Transactions on Intelligent Transportation Systems*, 22(1):23–35, 2019.

[81] You-Dong Liang and Brian A. Barsky. A new concept and method for line clipping. *ACM Transactions on Graphics (TOG)*, 3(1):1–22, 1984.

[82] Saeed Ghoorchian and Setareh Maghsudi. Multi-armed bandit for energy-efficient and delay-sensitive edge computing in dynamic networks with uncertainty. *IEEE Transactions on Cognitive Communications and Networking*, 7(1):279–293, 2020.

[83] Aleksandrs Slivkins et al. Introduction to multi-armed bandits. *Foundations and Trends in Machine Learning*, 12(1-2):1–286, 2019.

[84] Liangkai Liu, Sidi Lu, Ren Zhong, Baofu Wu, Yongtao Yao, Qingyang Zhang, and Weisong Shi. Computing systems for autonomous driving: State of the art and challenges. *IEEE Internet of Things Journal*, 8(8):6469–6486, 2020.

[85] Mouhamed Abdulla, Erik Steinmetz, and Henk Wymeersch. Vehicle-to-vehicle communications with urban intersection path loss models. In *2016 IEEE Globecom Workshops (GC Wkshps)*, pages 1–6. IEEE, 2016.

[86] Yuzhe Xu, Thaha Mohammed, Mario Di Francesco, and Carlo Fischione. Distributed assignment with load balancing for dnn inference at the edge. *IEEE Internet of Things Journal*, 10(2):1053–1065, 2022.

[87] Samuel Dodge and Lina Karam. Understanding how image quality affects deep neural networks. In *2016 eighth international conference on quality of multimedia experience (QoMEX)*, pages 1–6. IEEE, 2016.

[88] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[89] Jong-Chyi Su, Matheus Gadelha, Rui Wang, and Subhransu Maji. A deeper look at 3d shape classifiers. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, pages 0–0, 2018.

[90] Nehad Hameed Hussein, Chong Tak Yaw, Siaw Paw Koh, Sieh Kiong Tiong, and Kok Hen Chong. A comprehensive survey on vehicular networking: Communications, applications, challenges, and upcoming research directions. *IEEE Access*, 10:86127–86180, 2022.

[91] Joannes Vermorel and Mehryar Mohri. Multi-armed bandit algorithms and empirical evaluation. In *Machine Learning: ECML 2005: 16th European Conference on Machine Learning, Porto, Portugal, October 3-7, 2005. Proceedings 16*, pages 437–448. Springer, 2005.

[92] Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Return of the devil in the details: Delving deep into convolutional nets. *arXiv preprint arXiv:1405.3531*, 2014.