

Resource Allocation and Task Scheduling for Integrated Sensing and Communications

by

Shisheng Hu

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2025

© Shisheng Hu 2025

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Examiner: Qiang Ye
Professor
Faculty of Computer Science
Dalhousie University

Supervisor(s): Xuemin (Sherman) Shen
University Professor
Department of Electrical and Computer Engineering
University of Waterloo

Internal Member: Zhou Wang
Professor
Department of Electrical and Computer Engineering
University of Waterloo

Internal Member: Kshirasagar Naik
Professor
Department of Electrical and Computer Engineering
University of Waterloo

Internal-External Member: Xinzhi Liu
Professor
Department of Applied Mathematics
University of Waterloo

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

Integrated Sensing and Communications (ISAC) has emerged as a promising paradigm for future Sixth-Generation (6G) wireless networks. In this paradigm, wireless networks can have both Sensing and Communication (SAC) capabilities using shared network resources. ISAC not only enables the provisioning of SAC services but also has the potential to enhance their performance: end-user devices can offload sensing data collection tasks to Access Points (APs) co-located with edge servers and upload raw or preprocessed sensing data to powerful edge servers for high-performance processing; meanwhile, APs can leverage contextual information about communication tasks obtained through sensing, such as the visibility of virtual content in mobile Augmented Reality (AR) streaming, to enhance communication efficiency. The interesting issue is to efficiently utilize network resources to optimize SAC service performance in the presence of high and spatiotemporally varying service demands. However, the main technical challenges are: 1) how network resource allocation and SAC task scheduling are proactively determined to enable efficient coordination between APs and mobile end-user devices for achieving satisfactory service performance; 2) how an end-user device adaptively offloads computation-intensive Deep Neural Network (DNN)-based sensing tasks to an edge server to optimize task performance, under dynamic task arrival, task processing, and server workload statuses; and 3) how an AP efficiently acquires contextual information about communication tasks through sensing individual mobile AR users and dynamic environment for resource-efficient mobile AR streaming.

In this thesis, we develop efficient resource management schemes for ISAC, including resource allocation and task scheduling, to address the above three technical challenges. First, we investigate proactive resource management for ISAC, determining the reservation of radio and computing resources, the active probability of mobile devices for communications, and the partitioning of sensing regions. To cope with the non-stationary spatial distributions of mobile devices and sensing targets, which can result in the drift in modeling the distributions and ineffective resource management decisions, we construct Digital Twins (DTs) of the network slices for individual SAC services. In each DT, a drift-adaptive DNN-empowered statistical model and an emulation function are developed for the spatial distributions in the corresponding slice, which facilitates closed-form decision-making and efficient validation of a resource management decision, respectively. Numerical results demonstrate that the proposed scheme can significantly enhance service satisfaction ratios and reduce resource consumption compared to benchmark schemes. Second, we investigate task offloading for DNN-based sensing data processing. Particularly, we consider that an end-user device stochastically generates and adaptively offloads DNN-based sensing data processing tasks to an AP co-located with an edge server. To adapt to the

dynamic on-device and edge server workload status, leveraging the multi-layer and multi-exit architecture of the considered DNNs, an offloading decision for each sensing task is made on whether and when to stop on-device task processing and offload the task to the edge server to complete the processing. Two DTs are constructed to evaluate all potential offloading decisions for each sensing task, which provides augmented training data for a machine learning-assisted decision-making algorithm, and to estimate the task processing status at the device, which avoids frequently fetching the status information from the device and thus reduces the signaling overhead. Simulation results demonstrate the outstanding performance of the proposed task offloading scheme in terms of balancing sensing result accuracy, delay, and energy consumption. Third, we propose an efficient resource allocation scheme in sensing-assisted mobile AR streaming. In specific, we consider that the position and surrounding environment of an AR user can be captured via sensing to extract contextual information for AR streaming, i.e., the visibility of virtual content. The goal is to minimize the overall radio resource consumption for delivering virtual content visible to the AR user by properly determining the radio resource allocation for user positioning and environment mapping. To this end, we first develop a mathematical model to estimate the content visibility uncertainty and the content delivery resource consumption. We then generate a reference resource allocation decision that guides a deep reinforcement learning-based decision process to efficiently adapt to non-stationary user and environment dynamics. Trace-driven simulations demonstrate that the proposed scheme significantly reduces radio resource consumption for delivering virtual content visible to an individual AR user, compared to benchmark schemes.

In summary, we have proposed efficient resource management schemes for ISAC that optimize SAC service performance with efficient resource utilization and practical operational complexity. The research results from the thesis provide valuable insights into the design of scalable and adaptive ISAC systems that seamlessly unify sensing, communications, and intelligence in the future 6G.

Acknowledgements

First, I would like to express my most sincere and deepest gratitude to my supervisor, Professor Xuemin Shen, for his invaluable guidance and constant support during my Ph.D. study at the University of Waterloo. His insightful thoughts and constructive advice have guided me in developing critical thinking and research abilities. His patient instructions and continuous encouragement have supported me in embracing and tackling challenges. Without his guidance and support, this thesis would not have been possible. From Professor Shen, I have also learned a positive philosophy of life and the importance of dedication for happiness and success, which has been greatly helpful to me. Professor Shen is and will always be my role model. I feel very fortunate and honored to be his student.

I would also like to thank Professor Weihua Zhuang for her kind help and support in my Ph.D. study. Her invaluable suggestions had significantly enhanced the quality of my collaborative research. Her rigorous academic attitude has inspired me to do in-depth thinking on my research, which greatly benefits this thesis.

I would like to thank Professor Zhou Wang, Professor Kshirasagar Naik, Professor Xinzhi Liu, and Professor Qiang Ye for serving on my thesis examination committee. Their valuable questions, suggestions, and comments have helped me to enhance the quality of the thesis. I am sincerely grateful for their precious time and efforts devoted to this thesis.

In the past four years, I have gained valuable friendships and help from former and current members in our BBCR group. I would like to thank Prof. Jie Gao, Prof. Mushu Li, Prof. Conghao Zhou, Prof. Kaige Qu, Dr. Mingcheng He, Dr. Xiaodan Shao, Yingying Pei, Dr. Yannan Wei, Ruoxu Wang, Dr. Xinyu Huang, Zhixuan Tang, Yixiao Zhang, Yiren Zhou for their great help in my research and life. Special thanks go to Prof. Wen Wu, Prof. Huaqing Wu, Dr. Qihao Li, Dr. Haojun Yang, Prof. Xiufang Shi, and Prof. Yilong Hui for their kind help and support. I am also grateful for all the wonderful time spent in Canada with Prof. Dongxiao Liu, Prof. Cheng Huang, Prof. Liang Xue, Han Yin, Lingshuang Liu, Xue Qin, and all current and former BBCR members. Thank you for your great support and precious friendship.

Finally, I would like to thank my dearest mother, father, and other family members for their unconditional love and steadfast support in my Ph.D. study.

Shisheng Hu
July 20, 2025
Waterloo, Ontario, Canada

Dedication

The Ph.D. thesis is dedicated to my beloved parents, *Liqin Zhou* and *Yuanlun Hu*.

Table of Contents

Examining Committee Membership	ii
Author's Declaration	iii
Abstract	iv
Acknowledgements	vi
Dedication	vii
List of Figures	xiii
List of Tables	xv
List of Abbreviations	xvi
1 Introduction	1
1.1 Overview of the Integration of Sensing and Communications	1
1.2 ISAC Use Cases in Future 6G	3
1.3 Resource Management for ISAC	4
1.4 Research Motivation and Contributions	6
1.4.1 Challenges of Resource Management for ISAC	6
1.4.2 Proposed Approaches	8

1.4.3	Research Problems and Contributions	9
1.5	Thesis Outline	12
2	Background and Literature Review	13
2.1	Integrated Sensing and Communications	13
2.2	Resource Management for ISAC	15
2.2.1	Resource Allocation	15
2.2.2	Task Scheduling	17
2.3	Resource Management for 6G Wireless Networks	20
2.4	Summary and Discussions	22
3	Proactive Resource Management for Integrated Sensing and Communi- cation	23
3.1	Background and Motivation	23
3.2	System Model and Problem Formulation	26
3.2.1	Considered Scenario	26
3.2.2	Slicing Scheme for Cooperative ISAC	27
3.2.3	Communication Model	29
3.2.4	Sensing Model	31
3.2.5	Problem Formulation	34
3.3	Network Slice DTs for Spatial Modeling and Network Planning	36
3.3.1	Spatial Distribution Data Collection	37
3.3.2	Statistical Spatial Modeling Function	37
3.3.3	Network Emulation Function	40
3.4	DT-Based Network Planning Algorithm	41
3.4.1	Statistical Spatial Modeling-Based Decision Making	41
3.4.2	Network Emulation-Based Decision Evaluation	44
3.4.3	Overall DT-Based Network Planning Algorithm	45

3.5	Simulation Results	45
3.5.1	Simulation Settings and Benchmarks	45
3.5.2	Effectiveness of DT-Based Network Planning Algorithm	47
3.5.3	Effectiveness of Model Update	49
3.5.4	Effectiveness of Model Ensemble	49
3.6	Conclusion	52
4	Task Offloading for DNN-Based Sensing Data Processing	53
4.1	Background and Motivation	53
4.2	System Model	56
4.2.1	Task Generation Model	56
4.2.2	Task Computing Model	56
4.2.3	Task Queuing Model	58
4.2.4	Task Utility	59
4.3	DT-Assisted Adaptive Device-Edge Collaboration on DNN Inference	62
4.3.1	Data for Establishing DTs	62
4.3.2	DT of On-Device DNN Inference	62
4.3.3	DT of Computing Workload Evolution	63
4.3.4	Analysis of DTs' Overheads	64
4.3.5	DT-Assisted Approach to Adaptive Device-Edge Collaboration . . .	64
4.4	Problem Formulation and Transformation	66
4.4.1	Problem Formulation	66
4.4.2	Problem Transformation	66
4.5	DT and Learning-Assisted Algorithm for Offloading decision-making	69
4.5.1	DT and Learning-Assisted Offloading Decision-Making	70
4.5.2	DT-Assisted Training of ContValueNet	71
4.6	Offloading Decision Space Reduction	73
4.6.1	DNN Layers with Negligible Execution Time	73

4.6.2	Properties of Workload Evolution	74
4.7	Simulation Results	77
4.7.1	Simulation Settings and Benchmarks	77
4.7.2	Adaptiveness to Dynamic Computing Workload	79
4.7.3	Effectiveness of DT-Assisted Training Data Augmentation	80
4.7.4	Complexity Reduction by Decision Space Reduction	82
4.8	Conclusion	83
5	Resource Allocation in Sensing-Assisted AR Streaming	84
5.1	Introduction	84
5.2	System Model	85
5.2.1	AR Content Delivery Model	85
5.2.2	Sensing Model	86
5.3	Analysis and Problem Formulation	88
5.3.1	User Positioning and Environment Map Uncertainties	89
5.3.2	Content Visibility Uncertainty	91
5.3.3	Content Delivery Resource Consumption	92
5.3.4	Problem Formulation	92
5.4	Model-Assisted Learning for Resource Allocation	93
5.4.1	Action Space Reduction	93
5.4.2	DRL Algorithm	94
5.5	Simulation Results	95
5.6	Conclusion	96
6	Conclusions and Future Works	97
6.1	Main Research Contributions	97
6.2	Future Research Directions	98
	References	101

A Proof of Proposition 1	118
B Proof of Proposition 2	119
C Proof of Proposition 3	121
D Proof of Proposition 4	123
E Proof of Proposition 5	125
F Proof of Proposition 7	127
G Proof of Proposition 8	128

List of Figures

1.1	The illustration of resource management for ISAC.	10
3.1	The considered cooperative ISAC network.	26
3.2	RoI partitioning-based sensing target assignment.	28
3.3	Two-stage slicing-based resource management scheme.	29
3.4	The considered beam pattern of mobile devices and an illustration of the scenario where two mobile devices in the sensing mode interfere each other.	32
3.5	The DT of the sensing slice.	36
3.6	Values of the parameters μ_k^U and σ_k^U in the joint spatial modeling across planning windows.	38
3.7	Values of the parameters λ_k^I and λ_k^U in the independent spatial modeling across planning windows.	39
3.8	Performance evaluation with varying number of mobile devices per square kilometer.	47
3.9	Modeling error across planning windows.	49
3.10	Performance evaluation in the early planning windows after model drift occurs.	50
3.11	Performance evaluation under varying model drift frequency.	51
4.1	DNN-based sensing data processing model.	57
4.2	Task queuing model.	58
4.3	DT-Assisted approach to adaptive device-edge collaboration on DNN inference.	65
4.4	Decomposition of the on-device queuing delay T_n^{lq}	67

4.5	DT and learning-assisted algorithm for DNN task offloading decision-making.	70
4.6	Full-size DNN and shallow DNN.	77
4.7	(a) Average DNN task utility versus task generation rate. (b) Average DNN task utility versus edge processing load.	79
4.8	(a) Average delay; (b) Average accuracy; (c) Average energy consumption versus DNN task generation rate.	80
4.9	Number of collected training samples with and without DT-assisted training data augmentation.	81
4.10	Average DNN task utility with and without DT-assisted training data augmentation.	81
4.11	Training loss with and without DT-assisted training data augmentation. .	82
4.12	(a) Complexity and (b) Average DNN task utility with and without decision space reduction.	82
5.1	Sensing-assisted mobile AR streaming model.	86
5.2	Signal propagation and environment map models.	87
5.3	Content visibility estimation.	91
5.4	Average resource consumption per time interval.	96

List of Tables

3.1	Main simulation settings of the ISAC network	46
4.1	Main simulation settings in DNN-based sensing data processing	78
5.1	Main simulation settings in sensing-assisted mobile AR streaming	95

List of Abbreviations

3GPP	3rd Generation Partnership Project
6G	Sixth-Generation
AI	Artificial Intelligence
AIoT	Artificial Intelligence of Things
AP	Access Point
AR	Augmented Reality
CRLB	Cramér–Rao lower bound
DFRC	Dual-Function Radar Communications
DL	Deep Learning
DNN	Deep Neural Network
DRL	Deep Reinforcement Learning
DT	Digital Twin
EFIM	Equivalent Fisher Information Matrix
FCFS	First-Come-First-Serve
FoV	Field of View
GAI	Generative Artificial Intelligence
IMU	Inertial Measurement Unit
IoT	Internet of Things
ISAC	Integrated Sensing and Communications
ITU	International Telecommunication Union
LiDAR	Light Detection and Ranging
LoS	Line-of-Sight
MDP	Markov Decision Process
MEC	Mobile Edge Computing

MIMO	Multiple-Input Multiple-Output
NLoS	Non-Line-of-Sight
OFDM	Orthogonal Frequency Division Multiplexing
PMN	Perceptive Mobile Network
QoE	Quality of Experience
QoS	Quality of Service
RAN	Radio Access Network
RE	Resource Element
RoI	Regions of Interest
S&C	Sensing and Communication
VA	Virtual Anchor

Chapter 1

Introduction

Sensing refers to the process of acquiring information about the physical world, including detecting and tracking physical objects and mapping a physical environment, which is crucial in various Internet of Things (IoT) applications, such as smart factories. However, efficiently satisfying the sensing demand is challenging since low-cost sensors often suffer from limited reliability in complex environments, while high-end sensors incur significant computational overhead and hardware costs. To address the challenge, wireless communication networks, with growing advantages in sensing, are expected to be leveraged. In addition, sensing enables awareness of the communication user status and the propagation environment, which can be leveraged to improve communication efficiency. These mutual benefits motivate the development of Integrated Sensing and Communication (ISAC) as a revolutionary paradigm in sixth-generation (6G) wireless communication networks. In this chapter, we first provide an overview of the ISAC paradigm and highlight representative use cases. We then examine the challenges of resource management for realizing the ISAC paradigm, which motivates our research. Finally, we outline three key research problems investigated in this thesis.

1.1 Overview of the Integration of Sensing and Communications

As IoT systems evolve toward higher levels of automation and intelligence, they require not only efficient data transmission but also autonomous acquisition of environmental information [1]. For example, for industrial automation, inspection robots can be deployed

to detect physical defects in infrastructure components and promptly relay the information to a control center for a timely and effective response [2]. The process of obtaining environmental information is known as sensing. Tasks in sensing can be broadly classified into three types: detection, estimation, and recognition [3]. Detection involves determining the presence/absence of objects of interest; estimation refers to extracting the parameters of a detected object, such as position and velocity; and recognition focuses on understanding what the object is and what it is doing, including aspects such as its type, identity, physical integrity, and activity.

Sensing is typically subject to strict requirements to guarantee the reliable operation of IoT systems. For example, for human and vehicle detection in industrial environments, an object positioning accuracy of 0.5 meters and a position update rate of 100 milliseconds are required [4]. Fulfilling these strict requirements in a cost-effective and efficient manner is challenging. In particular, low-cost sensors such as cameras and Inertial Measurement Units (IMUs) may yield low-quality sensing data under adverse conditions, such as low-light environments or rapid user motion [5]. Although high-end sensors, such as Light Detection and Ranging (LiDAR) sensors, can produce high-quality sensing data under adverse conditions, the resulting data is often voluminous, e.g., as high as 2.8 million data points per second, and demands computationally intensive processing [6].

To address the issue, the ISAC paradigm, where wireless communication networks are used to provide both Sensing and Communication (SAC) services, offers significant potential [3]. Fundamentally, wireless signals interact with environmental objects during propagation, which results in environmental information being encoded in the signals. Recently, the advancements initially aimed at enhancing communication performance have further improved the ability of wireless communication networks to acquire environmental information. Particularly, the use of large-scale antenna arrays and wide-band, high-frequency spectrum provides signals with high resolution in the angle, delay, and Doppler domains [7]. In addition, the integration of multi-tier computing resources into wireless networks enables advanced computation-intensive Artificial Intelligence (AI)-based processing of sensing data [7]. Furthermore, the dense distribution of network infrastructure and mobile devices improves sensing service accessibility and enables collaborative sensing from multiple viewpoints [8].

Notably, in the ISAC paradigm, wireless communication networks not only support sensing but also benefit from it [3]. First, by providing the new sensing service, network infrastructure and radio spectrum can be more efficiently utilized, which increases the revenue potential of network operators. Second, environmental information obtained through sensing can help optimize the performance of communication services. For example, information on the position and velocity of communication users can be used to steer beams

accurately toward the users [9]. In addition, information on user position and environmental objects can be used to identify virtual content within the user’s field of view (FoV), which facilitates resource-efficient content delivery [10].

Beyond academia, standardization organizations and industry have also demonstrated strong interest in the ISAC paradigm. The 3rd Generation Partnership Project (3GPP) studied ISAC in Release 19, identifying key usage scenarios and service requirements [4]. In November 2023, the International Telecommunication Union (ITU) included ISAC as one of the six key usage scenarios anticipated in 6G networks [11]. In February 2025, the 3GPP SA2 working group proposed six work tasks for the standardization of ISAC in the upcoming Release 20 [12]. The industry has conducted real-world experiments to evaluate the performance of ISAC. In March 2022, Huawei launched an ISAC system prototype demonstrating the feasibility of the ISAC paradigm [13]. In November 2024, China Unicom and ZTE jointly conducted a real-world trial of ISAC systems to support medical content delivery services [14].

Overall, ISAC is regarded as a promising paradigm for joint SAC service provision, offering significant potential to bring substantial social and economic benefits in future 6G wireless communication networks. This thesis focuses on developing efficient and adaptive resource management schemes to realize the potential of the ISAC paradigm.

1.2 ISAC Use Cases in Future 6G

ISAC use cases in the future 6G can be broadly divided based on whether sensing or communication functionality is the primary focus. For communication-assisted sensing, 3GPP Technical Report (TR) 22.837 examined how ISAC can enhance sensing capabilities across various industry sectors, including smart homes, smart factories, and low-altitude aerial operations [4]. For sensing-assisted communication, ISAC is expected to play a crucial role in improving communication performance in the future 6G, where channel conditions [9] and communication content [15] can be highly dependent on user mobility and environmental dynamics. In this thesis, we focus on one representative use case from each of these two types, including human-centric smart factory and Augmented Reality (AR) streaming.

- **ISAC for human-centric smart manufacturing:** Smart manufacturing in the Industry 5.0 era is expected to be human-centric, with machines assisting human workers in collaborative decision-making and task execution [16, 17]. To achieve this

goal, sensing plays a crucial role. Particularly, the detection of human workers and mobile robots is essential for maintaining safety and enabling coordinated interaction in shared workspaces [4], while their identification and activity recognition are essential for informed task execution [17]. However, relying solely on mobile robots for sensing may lead to degraded performance due to their limited ability to collect high-quality sensing data under adverse conditions and to process sensing data using high-accuracy yet computation-intensive models. The ISAC paradigm can be adopted to support sensing in this case. Particularly, sensing data collection tasks can be distributed between mobile robots and wireless infrastructure such as Access Points (APs), based on their respective sensing advantages. Additionally, sensing data processing tasks can be partially or fully offloaded to APs co-located with Mobile Edge Computing (MEC) servers to support high-performance processing.

- **ISAC for AR streaming:** Mobile AR allows users to engage with virtual content that is seamlessly integrated into the physical world, which empowers immersive experiences in applications such as gaming, learning, and manufacturing [18]. In AR streaming, an AR device first collects sensing data and sensing results related to the physical world, such as the user’s position, orientation, and environmental object detection and recognition. Then, the corresponding raw or rendered virtual content is delivered from a network server to the device, which is displayed to the user. By offloading content storage and processing to the network, AR streaming reduces the burden on AR devices but may incur considerable communication overhead. Specifically, a single frame of high-fidelity point cloud content can reach 96 Mb, requiring 2.9 Gbps to support 30-frames-per-second streaming [19]. Additionally, on-device sensing can yield inaccurate results under adverse conditions, necessitating the streaming of redundant virtual content [20, 21]. To tackle the challenge, the ISAC paradigm can be leveraged, where wireless communication infrastructure collaborates with AR devices to enable low-uncertainty environment sensing in complex scenarios, thereby empowering communication-efficient AR streaming [15].

1.3 Resource Management for ISAC

Resource management in wireless networks mainly involves two parts: resource allocation and task scheduling, which determine the capacity and performance of both SAC services under the ISAC paradigm. Considering resource limitation and utilization overhead, resource management is essential to efficiently meet the growing demands and stringent

performance requirements of SAC services. As follows, we introduce the two parts of resource management under the ISAC paradigm in detail.

- **Resource allocation:** The quantity of allocated resources directly influences the achievable capacity and performance of SAC services. First, in communication service provision, radio resource allocation determines the data transmission rate, latency, and reliability [22]. In sensing service provision, the allocation of multi-dimensional radio resources impacts the sensing performance in different aspects. For example, the spectrum bandwidth and time slots affect the resolution of range and velocity estimation, respectively [23], while the number of beams determines the maximum number of targets that can be simultaneously monitored [24]. Additionally, the allocation of edge computing resources determines the processing time of sensing data and constrains the complexity and accuracy of the algorithms used for processing the sensing data. For using radio resources for SAC service provision, there are primarily two kinds of approaches: unified waveform design and orthogonal resource allocation [3, 7, 25]. In the unified waveform design approach, the SAC functions are implemented using a single signal waveform. This approach may achieve a high resource utilization efficiency, but it introduces high complexity [7, 25]. In the orthogonal resource allocation approach, resources are individually allocated to transmit sensing and communication signals separately. Although the resource utilization efficiency may be lower, this approach is more straightforward to implement and thus receives considerable attention [23, 25, 26, 27, 28].
- **Task scheduling:** In a wireless network adopting the ISAC paradigm for SAC service provision, there exist three main types of tasks: communication tasks, sensing data collection tasks, and sensing data processing tasks. First, communication task scheduling determines the timing at which mobile devices and network infrastructure to transmit or process communication data packets. Second, the scheduling of sensing data collection tasks would additionally determine sensing task assignment to mobile devices and infrastructure. Heuristically, the task corresponding to a target can be assigned to the nearest device for minimal large-scale path loss in sensing signal propagation [29], while more sophisticated decision-making is required to jointly account for available radio and computing resources [30, 31]. Third, the scheduling of sensing data processing tasks can operate at a higher level of granularity by task decomposition and configuration. For example, a multi-layer Deep Neural Network (DNN)-based task can be decomposed into multiple subtasks, each involving data processing by one or more DNN layers [32]. Additionally, a multi-exit DNN allows for adaptive selection of the exit DNN layer to complete a task based on resource

status, such as channel conditions [33] and edge server workload [34].

In this thesis, we focus on investigating the above two key aspects of resource management in wireless networks to efficiently support satisfactory SAC service provision under the ISAC paradigm.

1.4 Research Motivation and Contributions

The ISAC paradigm holds significant promise for future 6G networks, where resource management plays a crucial role in optimizing the capacity and performance of SAC services, as well as the resource utilization efficiency. However, achieving resource management that satisfies service demands and performance requirements while minimizing resource consumption is challenging, which motivates the research in this thesis. In this section, we first identify the challenges of resource management for ISAC. Then, we introduce our approaches that have the potential to address the challenges. After that, we present three research contributions that address the challenges based on the approaches.

1.4.1 Challenges of Resource Management for ISAC

There are several critical challenges of resource management for ISAC, given as follows:

- **Dynamic service demand:** Time-varying and high-dimensional user and environmental contexts jointly drive complex dynamics in SAC service demand. First, spatial distributions of mobile devices and sensing targets, which request the services, are time-varying due to their mobility. In addition, the resource demand of individual devices or targets can be time-varying. In AR streaming, a device requests the virtual content corresponding to the real-time surrounding environment, whose data size is content-specific [18]; for monitoring a sensing target, the collection frequency and processing complexity of sensing data depend on the predictability and the complexity of the target's motion [23, 35]. The dynamic service demand poses challenges for efficient and effective resource management. Tractable mathematical models, such as Poisson processes, provide analytical characterizations of service demand distributions. Then, resource management decisions with quantifiable confidence regarding demand satisfaction can be efficiently derived. Nevertheless, such tractability comes at the cost of limited modeling capacity, making it difficult to capture high-dimensional and nonlinear dynamics in practice. Deep Learning (DL)

models with a large number of parameters and flexible architectures can effectively capture the joint effects of user context, environmental conditions on the temporal variation of service demand. Deep Reinforcement Learning (DRL) models implicitly characterize the effects for resource management and obtain the optimal policy in a trial-and-error manner. However, these models typically produce deterministic predictions without quantifiable probabilistic information, which limits the reliability of both the prediction results and the resource management decisions based on them. In addition, these models are highly data-driven, requiring training on data with similar statistical properties to those of the current input for accurate prediction. When data drift occurs, i.e., the statistical properties change due to shifts in user behavior and environmental conditions, the prediction accuracy will be degraded. Collecting sufficient data for model update may take considerable time, during which the resulting resource management decisions can be unreliable.

- **Limited network resources:** Sensing services are typically subject to substantial demand and stringent performance requirements, leading to considerable resource demand. Nevertheless, wireless network resources, such as radio spectrum and computing, are inherently constrained. As a result, sensing service provision under the ISAC paradigm would reduce the resource availability for communication service provision and vice versa. For the satisfaction of both services, resource management should achieve improved resource utilization efficiency. In addition, network resources are associated with operational costs that scale with their utilization. Therefore, even if resources are sufficient, the overall resource consumption needs to be minimized for maximizing network operators' revenues. The implementation of sensing function can be beneficial in enhancing the resource utilization efficiency of communication service provision, e.g., localizing the scatters/reflectors in user surrounding environment can provide alternative signal beamforming directions, and further enhances data transmission efficiency [36]. However, the relation between the benefit and the sensing resource consumption can be complex and dynamic due to the time-varying user context, e.g., transmission data rate requirement, and environmental factors, e.g., moving scatters/reflectors.
- **Complex decision making:** Given the intrinsic coupling between sensing and communication service provision under the ISAC paradigm, joint resource management is beneficial for achieving optimal resource utilization efficiency and service performance. However, executing either sensing or communication tasks involves multi-dimensional resources available at diverse infrastructure and devices and is associated with multiple performance metrics. Consequently, the joint resource man-

agement problem results in a high-dimensional optimization objective and a large set of decision variables, which poses significant challenges for algorithm design. Furthermore, SAC tasks are typically generated asynchronously, leading to a multi-timescale nature of the joint problem with further increased complexity. One potential solution is to decompose the joint optimization problem into subproblems across different services and timescales. While this decomposition can reduce computational complexity, it may come at the cost of degraded optimality. To mitigate this degradation, it is essential to characterize and exploit the interdependence between subproblems during the decomposition effectively. Nevertheless, capturing such coupling across timescales often requires temporal prediction, which is particularly difficult when the prediction window is long and the service demand is non-stationary. In addition, the coupling among subproblems corresponding to different tasks is often complex and difficult to express in a tractable form.

1.4.2 Proposed Approaches

To address the challenges, we develop resource management schemes for ISAC based on the approaches given as follows:

- **Slicing-Based two-stage scheme:** With the virtualization of physical infrastructure and resources, network slicing allows for creating virtual networks, referred to as network slices, that support different services [37]. Network slicing offers enhanced flexibility in resource management to support ISAC. We propose a two-stage slicing-based resource management scheme for ISAC. Over a large timescale, individual network slices are configured for SAC services through proactive resource management, aiming to statistically guarantee service satisfaction with minimal resource consumption. Then, small-timescale resource management for individual users and tasks corresponding to the SAC services can be performed separately or jointly to optimize the balance between the management complexity, resource utilization efficiency, and service performance.
- **Digital twins for ISAC:** Digital Twins (DTs) are envisioned as a promising paradigm for characterizing service demand and service provisioning capabilities in future 6G networks [38]. DTs can be created as configurable data representations and data processing functions for individual network entities, including communication users, sensing targets, infrastructure, network slices for SAC services, and SAC service provisioning processes. Serving as repositories for both raw and processed data, includ-

ing real-time and historical records, DTs enhance data availability and enable data-intensive, AI-based functions to characterize the information on the entities, such as resource utilization status and SAC service demand. DTs also support emulation functions that generate the information on the entities in hypothetical conditions, e.g., users' satisfaction of SAC services given a resource allocation decision, thereby providing additional data to facilitate resource management. Moreover, by integrating both data and functional modules, DTs enable mutual coordination between them to support resource management. Specifically, data requirements of functions guide data collection and provisioning, while data dynamics, such as temporal drift, trigger timely updates to the functions.

- **Hybrid data/model approaches to resource management:** Data- and model-driven approaches can complement each other to enable more efficient resource management for ISAC. Particularly, data-driven approaches can support accurate prediction of parameters in statistical models of SAC service demands, which facilitate the derivation of closed-form resource management solutions. In addition, model-driven approaches can support efficient evaluation of resource management decisions, such as estimating user satisfaction with SAC services. With the evaluation, ineffective and potentially optimal decisions can be identified to guide data-driven approaches in achieving an effective decision-making policy, e.g., reducing the DRL action space. As a result, the reliability and adaptability of data-driven approaches, especially under non-stationary conditions, can be enhanced.

1.4.3 Research Problems and Contributions

In this thesis, we aim to develop resource management schemes for ISAC that optimize SAC service performance with efficient utilization of network resources. To achieve this objective, we focus on the following three research problems.

- 1) We investigate proactive resource management for ISAC, as shown in Problem 1 in Fig. 1.1. In the considered industrial scenario, mobile devices, such as mobile robots, have communication demand and sensing demand to monitor sensing targets, e.g., human workers, in their sensing Regions of Interest (RoIs). Our objective is to efficiently realize the collaboration between Access Points (APs) and mobile devices in the provision of communication and sensing services, such that the minimum spectrum and edge computing resources are consumed in meeting the service demand. To achieve this objective, we create and configure separate network slices to provide

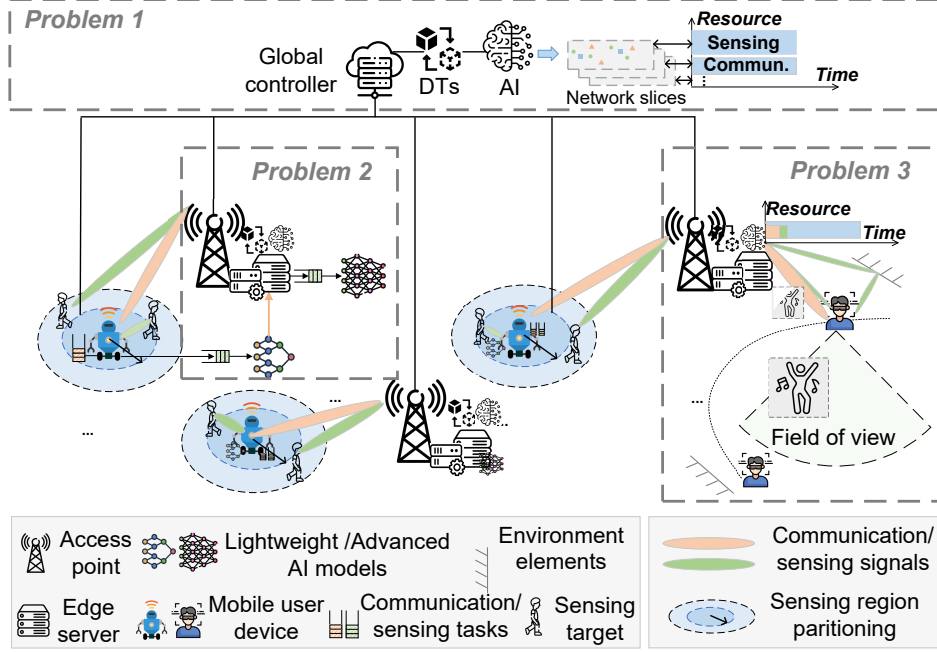


Fig. 1.1: The illustration of resource management for ISAC.

the communication and sensing services, respectively. Particularly, we determine the active probability of communications for mobile devices, partition the sensing RoI of each mobile device, and reserve network resources accordingly, empowering low-complexity probabilistic task scheduling, distance-based sensing target assignment, and resource allocation in small timescales. To cope with the non-stationary spatial distributions of mobile devices and sensing targets, which can result in the drift in modeling the distributions and ineffective resource management decisions, we construct DTs of the slices. In each DT, a drift-adaptive statistical model and an emulation function are developed for the spatial distributions in the corresponding slice, which facilitates closed-form decision-making and efficient validation of a resource management decision, respectively. Simulation results show that the proposed drift-adaptive slicing-based resource management scheme can increase the service satisfaction ratio and reduce resource consumption, compared with benchmark schemes.

- 2) We investigate task offloading for DNN-based sensing data processing, as shown in Problem 2 in Fig. 1.1. In the considered scenario, an end device stochastically generates DNN inference-based sensing data processing tasks. A full-size DNN is deployed

at an edge server for high-accuracy inference, and a shallow DNN, which consists of the first several layers of the full-size DNN concatenated by an exit branch, is deployed at an end device for low-latency on-device inference. A network controller makes the offloading decision for a sensing task, determining whether and when to stop the inference at the device and offload intermediate results to complete the inference at the edge server. Our objective is to optimize the performance of DNN inference-based sensing tasks in terms of balancing inference accuracy, delay, and energy consumption. To achieve this objective, a DT is constructed to estimate the inference status at the device, empowering multi-step adaptive task offloading. In addition, another DT is constructed to evaluate all potential scheduling decisions for each DNN inference task, which provides augmented training data for a machine learning-assisted task offloading algorithm. Leveraging mathematical model-based analysis, we derive necessary conditions for optimal offloading decisions to reduce the complexity of the offloading algorithm. Simulation results demonstrate the outstanding performance of our DT-assisted approach in terms of balancing the inference accuracy, delay, and energy consumption.

- 3) We investigate radio resource allocation in a sensing-assisted communication scenario, as shown in Problem 3 in Fig. 1.1. In the considered scenario, an AR user navigates a two-dimensional plane to view virtual content items associated with specific locations on the plane. Each item is presented as a volumetric video and cached at an edge server co-located with an AP that provides wireless coverage over the plane. Due to the limited FoV of the AR user and potentially large size of the content items, the AP estimates and delivers only the parts of the content items visible to the user. To reduce the uncertainty of content visibility, the AP transmits sensing signals to position the user and the environmental reflectors around the user, with the latter referred to as environment mapping. The objective is to minimize overall radio resource consumption for delivering virtual content items visible to the AR user by optimizing radio resource allocation for user positioning and environment mapping. To achieve this objective, we propose a novel model-assisted learning scheme for radio resource allocation. In specific, we first develop a mathematical model to estimate the content visibility uncertainty and the content delivery resource consumption. We then generate a reference resource allocation decision that guides a DRL-based decision process to efficiently adapt to non-stationary user and environment dynamics. We conduct trace-driven simulations to evaluate the performance of the proposed scheme, and the results demonstrate that the proposed scheme significantly reduces radio resource consumption for delivering virtual content visible to an individual AR user, compared to benchmark schemes.

1.5 Thesis Outline

The remainder of the thesis is organized as follows: In Chapter 2, we provide a comprehensive background and literature review of resource management for ISAC. In Chapter 3, we develop a proactive resource management scheme to ensure statistical satisfaction of SAC service requirements efficiently. The scheme captures the non-stationary spatial distributions of mobile devices and sensing targets, and adaptively optimizes resource reservation and sensing region partitioning. In Chapter 4, we investigate the offloading of DNN-based sensing data processing tasks and propose an adaptive approach to capturing task processing status and task arrival dynamics for optimizing task performance. In Chapter 5, we design a radio resource allocation scheme for adaptive user positioning and environment mapping that efficiently support content visibility-aware AR streaming. Finally, we conclude the thesis and discuss future research directions in Chapter 6.

Chapter 2

Background and Literature Review

In this chapter, we provide the background of ISAC, a survey of existing literature on the resource management for ISAC, and an introduction of emerging techniques that have the potential to empower efficient resource management for ISAC.

2.1 Integrated Sensing and Communications

SAC are two functionalities that both rely on signal transmission, yet serve fundamentally different purposes. Specifically, sensing focuses on acquiring and interpreting environmental information, while communication involves the transmission of information between two or more points. These two functionalities impose distinct requirements on signals, including signal waveform, directionality, and spectrum usage. Due to the limited capabilities of signal processing techniques, these distinct requirements could not be effectively fulfilled by a single system. As a result, the two functionalities were traditionally separated into distinct systems: radar and wireless communication networks.

The co-existence of SAC functionalities within a single system has gained significant attention in recent years. This co-existence is made possible by significant advances in wireless signal processing techniques, particularly Multiple-Input Multiple-Output (MIMO) [3], and is driven by increasing SAC demands from emerging applications. Due to the explosive growth in communication data traffic, researchers have explored the use of radar spectrum and infrastructure for communication, giving rise to the Dual-Function Radar Communications (DFRC) paradigm [39]. More recently, wireless communication networks have been rapidly evolving through the dense deployment of powerful infrastructure [1], while sensing demands have become increasingly prevalent across diverse and spatially distributed

environments, and sensing requirements have become increasingly strict to ensure a high level of automation in various Internet of Things (IoT) applications [40]. As a result, increasing attention has been directed toward embedding the sensing functionality into wireless communication networks, exemplified by the Perceptive Mobile Network (PMN) paradigm [41].

Beyond the co-existence, further integration of SAC functionalities in wireless networks is expected to explore their mutual benefits and co-designs, giving rise to the ISAC paradigm [3]. The two features of the ISAC paradigm, i.e., SAC mutual benefits and co-designs, are introduced as follows.

- **SAC mutual benefits:** By sensing surrounding environments, wireless networks may not only provide sensing services to fulfill ubiquitous sensing demands, but also enhance the provision of communication services. The enhancement occurs in two main aspects: better characterizing and fulfilling communication demands. Particularly, information about the physical environment helps identify the content that should be delivered to users, while location information on communication users empowers directional transmission of communication signals, thereby improving communication efficiency.
- **SAC co-designs:** Due to the mutual benefits and the sharing use of resources in wireless networks, the SAC functionalities in the ISAC paradigm need to be co-designed. Two common approaches to the co-design are unified waveform design and orthogonal resource allocation [3, 7, 25]. In the former, a single waveform is reused to simultaneously serve both functionalities, which achieves a high level of SAC integration and promises efficient resource utilization. In the latter, orthogonal resources are allocated to generate separate signals for the two functionalities, respectively. This approach offers a low design complexity and a high flexibility in signal waveform selection. In this thesis, we only focus on the orthogonal resource allocation approach.

By exploring the SAC mutual benefits and co-designs, ISAC promises efficient resource utilization of wireless networks to enhance sensing and communication performances. It empowers the seamless integration of physical and digital worlds, enabling a wide range of applications in 6G era, such as smart cities, industrial automation, and extended reality.

2.2 Resource Management for ISAC

In this section, we provide a comprehensive review of the two key parts in resource management for ISAC, including resource allocation and task scheduling.

2.2.1 Resource Allocation

In resource allocation for ISAC, the objective is to determine the quantity of resources for SAC functionalities, so that resources can be efficiently utilized to fulfill SAC demands and performance requirements. We review existing works of resource allocation for ISAC from the following different aspects.

Optimization problems with different structures have been studied under various conditions. In the ideal case, it is desirable to use the minimum amount of resources to fulfill the SAC demands with satisfactory performances [42, 43, 44] or efficiently leverage the residual resources to improve further the SAC performances [23, 31, 45, 46]. In [43], Zhang *et al.* investigated the configuration of beam resources to minimize the transmit power of network infrastructure while satisfying the target localization accuracy and communication data rate requirements. In [31], Zhao *et al.* investigated the allocation of radio and edge computing resources to individual users for optimizing SAC performances with constrained sensing time cost, guaranteed sensing data quality, and communication data rate. When there are substantial sensing or communication demands, resources in a wireless network may be insufficient to satisfy all the demands. In this case, optimization problems with two structures have been investigated primarily. First, when communication serves as the primary functionality of a wireless network, resources can be allocated to meet sensing demands on a best-effort basis, provided that communication demands are satisfied [28]. Second, considering the different utilities associated with sensing and communication services, e.g., their respective revenue contributions to network operators, optimization problems are formulated to jointly maximize the overall utility under only the resource constraint [47].

Sensing performances have been evaluated either indirectly or directly in the resource allocation for ISAC. In indirect evaluation, features associated with the wireless signals, such as signal-to-noise ratio [26, 42, 44], sampling rate [26, 27, 35], and mutual information [3, 31], are calculated and used to reflect sensing performances. The strong correlation between the features and various sensing performances guarantees a certain level of evaluation accuracy [3, 48], and the efficient calculation of such fixed features ensures low complexity and high generalization of resource allocation. In direct evaluation, closed-form

mathematical formulas are developed to characterize the relationship between resources and a specific sensing performance. To achieve this goal, estimation theory methods and function-fitting methods have been primarily used. Specifically, for parameter estimation, the Fisher information is calculated to evaluate the amount of information about certain parameters, such as the target location, that is embedded in sensing signals. This information is then used to derive a lower bound on the parameter estimation error, namely the Cramér–Rao Lower Bound (CRLB) [49], which is incorporated in guiding the resource allocation [23, 28, 31, 43, 45, 46]. For recognition tasks involving complex models such as DNNs, mathematical estimation of sensing performances is often intractable. To this end, fitting functions are usually used and optimized [50, 35, 51]. In [35], He *et al.* adopted a non-decreasing fitting function to characterize the relationship between the signal sampling rate and the classification accuracy of sensing target activities. In [50], Qu *et al.* developed a neural network-based fitting function to estimate object classification accuracy given multi-dimensional features of sensing data.

Resource allocation decisions have been designed to operate at different timescales. To adapt to dynamic service demand, such as task arrival [52], dynamic resource status such as channel conditions of radio spectrum [23], and spatial distributions of users and devices [25], it is beneficial to have a small timescale and high frequency to adjust resource allocation decisions. Information on the real-time spatial distributions empowers small-timescale resource allocation for ISAC, e.g., power control [28, 53] and spectrum allocation [23]. To reduce complexity, resource allocation decisions can be made at a large timescale. To achieve this goal, statistical spatial point process models that are temporally stationary, i.e., those with constant parameters over time, are commonly used to characterize the spatial distributions [54]. Accordingly, the formulas for estimating spatially averaged sensing and communication performances, e.g., target detection probability and ergodic transmission rate, can be derived for performance analysis and network planning [55, 56]. Spatial information is even more critical for resource allocation in cooperative ISAC networks. In cooperative ISAC networks, multiple devices can be used for monitoring individual sensing targets. The information about their spatial distributions, along with that of the sensing targets, determines the optimal coordination of the devices and the corresponding resource allocation. Time allocation was optimized in time-division-based ISAC systems, where APs and mobile devices cooperate in sensing service provision in a bi-static sensing mode [57].

Both short-term and long-term SAC performances have been considered in the resource allocation for ISAC. Optimizing short-term performance, such as that of a real-time object localization task, is usually more straightforward because the performance can be directly characterized based on instantaneous observation of system dynamics. Then, model-driven approaches can be developed for resource allocation, such as convex optimization [28,

45, 46]. Optimizing long-term performance can be more beneficial due to the potential correlation between consecutive tasks; for example, the previous localization results of an object can be used as prior knowledge for future localization of the same object. For the optimization, characterizing the future system dynamics is required, and model-driven or data-driven approaches have been developed. In [23], Chen *et al.* proposed a Markov model to characterize the state evolution of channel condition and develop a closed-form formula of object localization accuracy over a large timescale. In [9], Liu *et al.* proposed a deep learning model that implicitly captures the channel condition evolution from the historical records for maximizing SAC performances over a large timescale.

2.2.2 Task Scheduling

Task scheduling for ISAC determines how individual SAC tasks are completed, given allocated resources. For the scheduling of communication tasks, researchers have made significant efforts and contributions before the emergence of the ISAC paradigm [58, 59, 60, 61]. Particularly, in [58], Luan *et al.* analyzed the impact of vehicle mobility on the communication throughput and optimized the scheduling of communication packets. In [59, 60], Gao *et al.* proposed a medium access control protocol and learning-based scheduling algorithm to satisfy the diverse requirements of a massive number of devices with sporadic data traffic in industrial IoT. In the following, we review the literature on scheduling sensing tasks, including sensing data collection tasks and sensing data processing tasks.

Sensing data collection tasks

Sensing data collection plays a crucial role in obtaining and updating the information about environmental objects. We review the scheduling of sensing tasks in terms of data collection timing and data collector selection.

Typically, the information about the physical environment changes continuously and needs to be updated periodically. To update the information, collecting new sensing data from the environment is beneficial, but not strictly necessary. Particularly, the historical estimates of information, such as the location of a target, and the pattern of change in that information, including the velocity of the target, allow for the prediction of the information. In [62], Chen *et al.* proposed a learning-based framework to adaptively determine the timing of sensing data collection in ISAC systems, which balances sensing accuracy and sensing data collection overhead. For optimizing the balance, when collecting sensing data, choosing a proper sensing data sampling rate is also crucial [3, 35, 63]. In [63], Li

et. al determined the sensing data sampling rate to optimize the tradeoff between object recognition accuracy, sensing signal overhead, and communication throughput. Moreover, the timing for collecting individual sensing data samples is also important since the channel conditions of allocated radio spectrum resources are time-varying [26, 27]. In [26], Xie *et al.* proposed to identify the timing when the channel conditions between communications and the network infrastructure are adverse for scheduling sensing data sample collection, such that the utilization efficiency of radio spectrum resources can be optimized.

In wireless networks adopting the ISAC paradigm, a dense distribution of devices, including wireless devices and infrastructures, can be embedded with sensing functionalities. Given the same spectrum resources, devices may obtain sensing data about a target with varying quality due to differences in hardware capabilities, relative geometry to the target, and wireless channel conditions [51, 64, 65]. This makes the selection of sensing data collectors crucial for optimizing sensing performances. Since signal propagation loss increases with propagation distance, a heuristic approach is to assign the sensing task corresponding to a target to the nearest device [29]. More sophisticated selection approaches are required to jointly account for radio and computing resource availability and condition [30, 31, 64]. In addition, the device selection for monitoring multiple sensing targets can be jointly considered since neighboring targets may be sensed with the same sensing signals with higher resource utilization efficiency [64]. To further enhance sensing performance, multiple devices can be selected to collect sensing signals from different viewpoints, which are jointly processed to obtain the final sensing results [40, 50, 51, 64, 65, 66, 67, 68]. In [40], Li *et al.* proposed a cooperative sensing scheme that evaluates the sensing data quality of individual devices, predicts the demand for collaborative sensing, and facilitates a threshold-based scalable scheduling algorithm. In addition, statistical information of sensing demands, such as the spatial distributions of sensing targets, was captured for optimizing network configurations, such as the density of network infrastructures [69, 70] and the clustering of infrastructures for interference management [29].

Sensing data processing tasks

The collected sensing data must be processed to obtain the sensing results. To fulfill the computing demand in the processing, the MEC paradigm can be adopted. In this paradigm, extensive research efforts were made on the collaboration of devices and edge servers in data processing [71, 72, 73, 74, 75] and data caching [76, 77, 78] for efficient resource utilization and satisfactory task performance. Notably, for recognition tasks such as classifying object category [50], user satisfaction [79], and user contact [80], as well as downstream tasks like vehicle path planning [81], DNN models have demonstrated outstanding performance.

Nevertheless, DNN inference and training usually require substantial data volumes and high computational cost, which requires efficient scheduling of DNN-based sensing data processing tasks. In the following, we review task scheduling in terms of the offloading of DNN inference tasks and the distributed execution of DNN training tasks.

Some existing works have optimized the number of DNN inference tasks offloaded from mobile devices to the computing servers to minimize the overall delay in task completion [82, 83, 84]. Due to the limited computing capabilities, a lightweight DNN was executed by AIoT devices for task processing to reduce the delay of on-device DNN inference at the cost of inference accuracy [83, 84, 85]. In such a case, the offloading decision was made to optimize the tradeoff between DNN inference costs, in terms of delay, energy consumption, etc., and inference accuracy. In addition, features of the input data of DNN tasks, such as data quality [83, 86] and data similarity of consecutive DNN tasks [87], were evaluated in determining the offloading decisions of the DNN tasks. Moreover, in [85], Wu *et al.* jointly optimized the data sampling rate and DNN model selection to satisfy the diverse accuracy requirements of sensing data processing tasks. In [68, 88], Qu *et al.* investigated the offloading of multiple DNN inference tasks that are generated by multiple devices or consecutively generated by one device, to meet DNN inference accuracy requirements efficiently.

Other works have investigated the partitioning of individual DNN inference tasks to adapt to dynamic server workloads, leveraging the multi-layer architecture of a DNN model [32, 34, 89]. In [32], Kang *et al.* proposed DNN partitioning-based task scheduling scheme, where the partition layer to minimize the delay or energy consumption of a DNN task was dynamically selected based on the real-time workload at a computing server. Other works evaluated the on-device computing workload for DNN task partitioning [90, 91]. Specifically, considering periodical task generation at the device, Hu *et al.* maximized the throughput of the DNN inference while preventing the DNN tasks from experiencing on-device queuing delay [90]. For the case when the DNN task generation at the device is non-periodical, Song *et al.* proposed a DRL-based algorithm to dynamically select the partition layer [91]. The aforementioned works adapted to the network dynamics, e.g., server workload [32], on-device workload [91], and dynamic channel condition [92], by making offloading decisions for each DNN task upon its generation. To make the device-edge collaboration more adaptive, the offloading decision of each DNN task was adjusted during the on-device DNN inference based on the real-time estimation and the statistics of channel conditions [33].

DNN models require training on continuously collected sensing data to ensure a high inference accuracy. For efficient training and retraining, the federated learning scheme has been proposed recently [93, 94, 95, 96], where end devices utilize their local sensing

data to obtain updates of a DNN model and then exchange and aggregate these updates for completing the training. Considering the limited computing capabilities of mobile devices, the split federated learning scheme has been proposed [97, 98, 99], where a DNN model is partitioned into two parts that are respectively updated by mobile devices and edge servers. When the data used for training a DNN model diverges from that recently collected, the inference accuracy of the DNN model may decrease, requiring the retraining of the model [100, 101, 102]. In [100], Mullapudi *et al.* developed an adaptive retraining scheme that periodically uses a large DNN to label data and assess the small DNN's accuracy, triggering retraining when accuracy drops below a threshold. In [101], Khani *et al.* proposed a dynamic DNN model placement scheme where only the parameters in the model with gradients above a threshold were delivered to reduce communication overhead while maintaining accuracy. In [102], we developed an adaptive sensing data offloading and selection scheme to minimize the overall computation overhead of DNN inference and DNN retraining while meeting the DNN inference accuracy requirements.

2.3 Resource Management for 6G Wireless Networks

Outside the context of the ISAC paradigm, several advancements in resource management for future 6G wireless networks have been made in recent years. In the following, we review network slicing, digital twins, and hybrid data/model-driven approaches.

With the proliferation of user devices and application scenarios, it is expected that more diverse services with different service requirements will be requested in wireless networks. Network slicing has emerged as an efficient solution to the multi-service provision [37]. Empowered by software-defined networking and network function virtualization, network slicing enables the creation of multiple logical networks on a single physical network to support diverse services. In the slicing for core networks, extensive research efforts have been made on the deployment and resource allocation of service function chains [103, 103]. In [104], Qu *et al.* proposed an adaptive learning-based resource allocation scheme, where the change point of service traffic statistics is detected and a DRL-based algorithm is developed for the adjustment of resource allocation. In the slicing for Radio Access Networks (RANs), extensive research efforts are made on resource reservation for individual services and resource allocation to individual users. In [105, 106], learning-based dynamic resource allocation schemes were proposed to achieve load balancing in consideration of vehicle mobility. In [22, 107], RAN slicing in two-tier networks was investigated, where the service regions and resource allocation were adaptively adjusted for the two-tier network infrastructures with different capabilities. In [108], Li *et al.* proposed an AI-driven RAN

slicing framework where data management was optimized for ensuring the performance of data-hungry AI algorithms. In [109], M. A. Hossain *et al.* proposed a RAN slicing scheme for ISAC, where the latency of individual SAC tasks was modeled and jointly minimized by adaptive resource allocation.

Network slicing empowers the provision of diverse services by the virtualization of resources and functions in wireless networks. Nevertheless, the efficient service provision requires accurately characterizing service provision status, including service demand, resource utilization, and user satisfaction. To achieve this goal, further virtualization of 6G wireless networks is desired to provide valuable and sufficient data for monitoring, analyzing, and predicting the status of service provision, and adaptively managing wireless resources [38, 110]. DTs are envisioned as a promising solution for the virtualization. The concept of DTs was originally proposed for product life-cycle management to achieve continuous monitoring and coordinated management of products [111]. In the context of wireless network management, a DT can be constructed as an assembly of data and data processing functions about individual network entities, including users, infrastructure, network slices, and the service provisioning process [38, 110]. Specifically, existing works have investigated establishing (i) user-level DTs to characterize the real-time user status and service demand, e.g., task arrival pattern in mobile edge computing applications [112], sensing devices' data quality in collaborative sensing scenarios [40], sensing demand of mobile users [73], channel condition variation [113], or (ii) network-level DTs to represent the real-time network-level status using proper data structures such as graphs, and to evaluate service provision using network simulators [114, 115] or learning-based algorithms [116, 117].

DTs ensure the provision of sufficient and valuable data to support adaptive resource management, particularly the powerful data-driven DL approaches [38]. Nevertheless, due to the large scale and high dynamics of wireless networks, large amounts of data need to be collected and processed to maintain the effectiveness of data-driven approaches, which compromises the efficiency of the approaches. To address this issue, mathematical models can be leveraged jointly. The benefits of mathematical model-driven approaches are mainly twofold. First, mathematical models are typically characterized by coarse statistical information, such as the arrival rate of SAC tasks, without requiring detailed information, such as the fine-grained task arrival pattern. Obtaining such coarse information requires a significantly smaller amount of data, and the resulting mathematical models are robust to variations in the detailed information. Second, with mathematical models, we can utilize probability theory to derive closed-form formulas for estimating and predicting network status, which can be used by mathematical optimization approaches to find effective resource allocation decisions efficiently. In [38], several options for hybrid

data/model-driven approaches were identified, including solution backup and refinement, as well as task division. In [107], Zhou *et al.* leveraged unsupervised learning to identify possibly effective resource allocation decisions from historical records, and evaluated the decisions by mathematical models for selection. In [40], Li *et al.* developed a sensing data quality threshold-based sensing demand prediction model and a threshold-based sensing scheduling model, where both thresholds were determined with the assistance of learning-based approaches. In [50], Qu *et al.* proposed a model-assisted learning framework for resource allocation in cooperative sensing, where deep reinforcement learning was used to make optimal cooperation decisions, and model-based optimization was used to obtain a corresponding resource allocation decision efficiently.

2.4 Summary and Discussions

In this chapter, we have provided a background on ISAC and a comprehensive review of the literature on resource management for wireless networks with and without the adoption of the ISAC paradigm. However, several crucial resource management problems in ISAC remain, which the approaches in existing works cannot efficiently address.

First, considering the large number of mobile devices and network infrastructures, coordinating them for SAC service provision involves substantial signaling overhead and computational complexity in real-time. This makes the proactive resource management, such as sensing region partitioning and resource reservation, beneficial, and requires an accurate spatial modeling of the devices along with the sensing targets.

Second, the generation of sensing data processing tasks and server workload can be highly dynamic in real-time. To optimize task performance in such cases, there is a need for flexible task execution models, efficient workload monitoring and emulation approaches, and adaptive task scheduling algorithms.

Third, sensing can be used to map a communication user's surrounding environment, facilitating the user's positioning in Non-Line-of-Sight (NLoS) conditions within user location-dependent AR streaming scenarios. Considering the non-stationary user mobility, adaptively balancing the trade-off between the resource overhead and communication efficiency due to environment mapping remains a challenge and deserves further investigation.

Chapter 3

Proactive Resource Management for Integrated Sensing and Communication

In this chapter, we propose a novel drift-adaptive proactive resource management scheme for ISAC. Particularly, we establish two network slices to provide SAC services, respectively. In the large-timescale planning for the slices, we determine the active probability for communications of mobile devices, partition the sensing RoIs of the devices, and reserve network resources accordingly, such that minimum resources are consumed in statistically guaranteeing SAC service performance. To cope with the non-stationary spatial distributions of mobile devices and sensing targets, which can result in the drift in modeling the distributions and ineffective resource management decisions, we construct DTs of the slices. In each DT, a drift-adaptive statistical model and an emulation function are developed for the spatial distributions in the corresponding slice, which facilitates closed-form decision-making and efficient validation of a resource management decision, respectively. Numerical results demonstrate that the proposed scheme can significantly enhance the service satisfaction ratio and reduce resource consumption compared to benchmark schemes.

3.1 Background and Motivation

ISAC is recognized as a transformative paradigm in the forthcoming 6G era [3]. However, the substantial resource demand and strict performance requirements of ISAC pose

a challenge to its implementation. Cooperative ISAC [66, 118], also known as networked ISAC [65, 119], is promising to tackle this challenge. In cooperative ISAC networks, multiple devices, each equipped with both SAC functions, can be used for SAC service provision. Particularly, multiple devices can be selected to monitor a sensing target, exploiting their respective strengths, such as the proximity of mobile devices to the target or the abundant resources available at APs. To handle the potential high complexity of multi-device and multi-service coordination in cooperative ISAC networks, slicing-based resource management, particularly, the two-stage slicing scheme [73, 106], can be exploited. Over large timescales, referred to as planning windows, network slices for the individual services can be proactively configured through network resource reservation, a process referred to as network planning [73], which reduces the complexity of small-timescale resource management for individual slices.

Despite the potential, the challenge of leveraging the slicing-based resource management scheme for cooperative ISAC networks is to efficiently and accurately characterize the random spatial distributions of mobile devices and targets. A statistical spatial model, such as a spatial point process, is commonly adopted due to its capability to derive closed-form estimation of the expected service demands and capacities for network planning [29, 57, 69, 70]. However, in temporally non-stationary environments, across planning windows, the parameters in a statistical spatial model that best fit the actual spatial distributions would change. If the change is drastic, the parameter prediction for the subsequent planning window may suffer from severe accuracy degradation [120], which causes the *drift* of the spatial model. Especially, a spatial model with finer granularity, defined as the level of detail or spatial resolution at which spatial distribution information is captured, would generally face a higher risk of the drift because the model is typically characterized by more parameters. For example, a cluster point process can be used to characterize the spatial correlation between mobile devices and targets [57], which makes it more accurate in most cases but less robust to a sudden change of the spatial correlation, compared with the use of two independent point processes [29, 69, 70].

DTs have received increasing attention for modeling and managing wireless networks. DTs, as digital representations of individual network entities or network slices, can help facilitate configurable collection and processing of their data and efficient and accurate modeling of them [38]. Particularly, introducing DTs of SAC slices for spatial modeling in cooperative ISAC networks has the following advantages to tackle the drift issue:

- *Efficient Data Utilization:* Through sensing service provision, the mobile devices and APs continuously obtain sensing data of sensing targets. Such sensing data, along with the location data of mobile devices, can be stored in the slice DTs and used for

spatial modeling.

- *Statistical Model Adaptation:* The collected data in the DTs can be used to continuously optimize the statistical modeling of the spatial distributions, such as the model granularity determination and the model parameter prediction.
- *Emulation-Based Model Validation:* Due to temporal correlation, historical snapshots of the spatial distributions can be used to emulate the distributions in the following planning window. Such emulation can be used to validate whether a statistical spatial model has adapted to the drift that occurred earlier.

In this chapter, we propose a drift-adaptive slicing-based resource management scheme for cooperative ISAC networks. Our objective is to efficiently realize the collaboration of APs and mobile devices in the SAC service provision, such that the minimum spectrum and edge computing resource is consumed in meeting the service demands. To achieve this objective, we propose a slicing-based resource management scheme, where separate network slices are created and managed to provide the SAC services, respectively. In the large-timescale planning of the slices, the sensing RoI of each mobile device is partitioned for low-complexity distance-based sensing target assignment in small timescales. Then, we formulate an optimization problem that minimizes the resource consumption in service provision by properly determining the sensing RoI partitioning and the resource reservation for the slices. By collecting and processing the historical spatial distributions of mobile devices and targets, we construct DTs of the network slices to obtain drift-adaptive statistical models and emulation functions for their spatial distributions in individual slices. Leveraging the statistical spatial models, we derive closed-form formulas in solving the optimization problem. In addition, via emulation, the reliability of a statistical spatial model and the corresponding network planning decision are validated. The main contributions of this paper are summarized as follows.

- We propose a slicing-based resource management scheme that enables low-complexity coordination of mobile devices and APs within ISAC networks by large-timescale network planning including sensing RoI partitioning and resource reservation.
- We construct DTs of network slices that provide drift-adaptive statistical models and emulation functions for characterizing the spatial distributions of sensing targets and mobile devices. Based on the DTs, we derive a closed-form and reliable network planning solution.

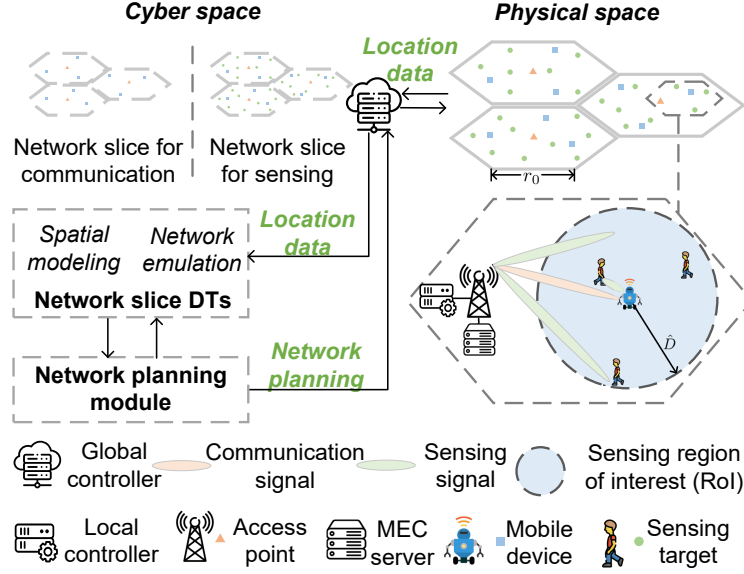


Fig. 3.1: The considered cooperative ISAC network.

The remainder of this chapter is organized as follows. In Chapter 3.2, we present the considered scenario, our proposed slicing-based resource management scheme for cooperative ISAC networks, and the network planning problem formulation. The establishment of DTs of network slices is introduced in Chapter 3.3. The network planning algorithm based on the slice DTs is introduced in Chapter 3.4. Simulation results are provided in Chapter 3.5, followed by the conclusion of this paper in Chapter 3.6.

3.2 System Model and Problem Formulation

3.2.1 Considered Scenario

As shown in Fig. 3.1, we consider an industrial zone, which is segmented to L non-overlapping service provision areas. Each area is a hexagon with a side length r_0 and a size of $A_0 = 3\sqrt{3}r_0^2/2$. In each area, there exist sensing targets, mobile devices, and an AP. The service demands and capabilities of them are introduced as follows.

- **Mobile Devices:** such as load-carrying and inspection robots, that have sensing and communication demands and capabilities within the ISAC paradigm [121]. Each

mobile device monitors the targets in their sensing RoI which is a circle centered at it with radius $\hat{D}$. In addition, the mobile devices have on-device computing resources that can be used for sensing data processing.

- **Sensing Targets:** such as human workers, that need to be detected to enhance workspace safety and efficiency. In addition, their state, such as the location, velocity, and activity, needs to be periodically tracked.
- **AP:** located in the center of the area, that has sensing and communication capabilities and has access to edge computing resources that can be used for sensing data processing.

We consider a cooperative ISAC network, where mobile devices and APs collaborate in sensing and communication service provision. In this network, the respective advantages of mobile devices and APs in sensing, such as closer proximity of mobile devices to sensing targets and more abundant resources available at APs, are leveraged for effective target assignment. In addition, the scheduling of mobile devices to perform either communication or sensing is optimized under a time-division-based ISAC paradigm, which ensures that the mobile devices can complete the sensing tasks for the assigned targets while completing their data transmission tasks [26]. With such cooperation, the network resources can be efficiently leveraged in the service provision.

3.2.2 Slicing Scheme for Cooperative ISAC

We propose the following two-stage slicing-based resource management scheme to efficiently realize the cooperation of mobile device and APs in the cooperative ISAC network.

RoI Partitioning-Based Target Assignment and Mono-static Sensing

Each AP collaborates with each mobile device in its coverage for monitoring the targets in the sensing RoI of the mobile device. First, the AP periodically detects targets that newly arrive or become active within the RoI of each mobile device. Then, a detected target is assigned to either an AP or a mobile device for tracking according to an RoI partitioning-based rule. Specifically, as shown in Fig. 3.2, we draw a concentric inner circle with radius $D_{\max}^{\text{L,U}}$ in the RoI of each mobile device and define it as the *on-device sensing subregion*. Due to the proximity, we assign the targets in the on-device sensing subregions to mobile devices and allocate shared spectrum resources for them to track the targets. Specifically,

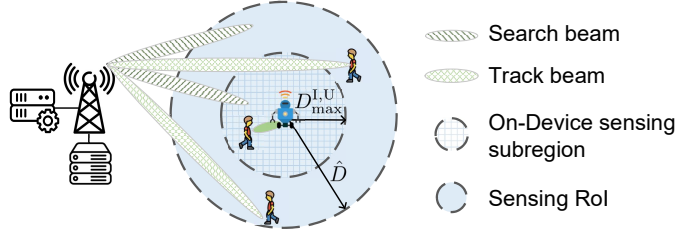


Fig. 3.2: RoI partitioning-based sensing target assignment.

a target will be assigned to the mobile device if it is (i) in the on-device sensing subregion of the mobile device and (ii) closest in distance to the mobile device than to any other mobile devices. For each assigned target, mono-static sensing is conducted. Specifically, a device or an AP transmits sensing signals towards the target, collects the signals that are reflected by the target, and estimates the state of the target based on the reflected signals.

Two-Stage Slicing-Based Resource Management

To efficiently provide the SAC services, we propose a two-stage slicing-based resource management scheme for ISAC. Particularly, two network slices are established by a global controller to provide SAC services, respectively. As shown in Fig. 3.3, the management of the network slices involves small-timescale network operation by local controllers at the APs and large-timescale network planning by the global controller:

- *Small-Timescale Network Operation:* At the beginning of each sensing interval that consists of T time slots, the above-mentioned RoI partitioning-based target assignment is carried out. At the beginning of each time slot that lasts τ seconds, each mobile device in the coverage of an AP is scheduled by the local controller at the AP to perform either mono-static sensing or communication according to pre-determined probabilities¹, to be detailed in Section 3.2.4.
- *Large-Timescale Network Planning:* At the beginning of each planning window that consists of M sensing intervals, the global controller makes a network planning decision including resource reservation and RoI partitioning. Specifically, spectrum

¹The scheduling can also incorporate the instantaneous channel conditions. For example, according to [26, 122], the optimal communication scheduling based on channel conditions has a threshold property. Given the active probability and the statistical property of the channel conditions, the threshold can be adjusted to ensure that the proportion of slots when the device is active for communication is approximately equal to the active probability.

and edge computing resources are reserved, and the probabilities for a mobile device to perform SAC in each time slot are determined. In addition, the radius of the on-device sensing subregions is determined.

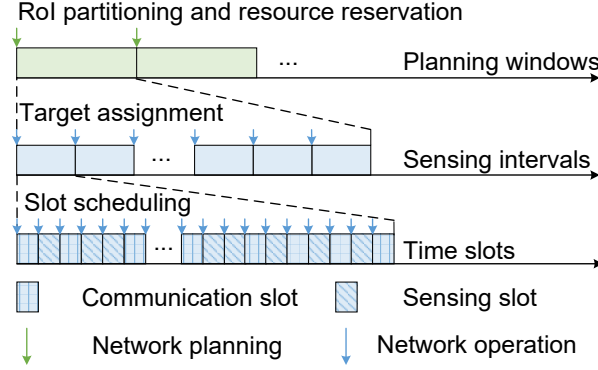


Fig. 3.3: Two-stage slicing-based resource management scheme.

In the proposed two-stage resource management scheme, large-timescale network planning empowers low-complexity real-time sensing target assignment and resource allocation for individual devices, while small-timescale network operation incorporates real-time network dynamics, such as instantaneous channel conditions, to further improve service provision performance and resource utilization efficiency. In this paper, we focus on large-timescale network planning, taking into account the varying spatial distributions of mobile devices and targets across different planning windows. As shown in Fig. 3.1, the global controller aggregates historical spatial distributions of mobile devices and targets in each network slice to establish DTs of the network slices. The DTs are used to achieve adaptive spatial modeling and network emulation, to be detailed in Section 3.3. Based on the DTs, a network planning algorithm is developed to efficiently and reliably make the network planning decision, which will be detailed in Section 3.4.

3.2.3 Communication Model

The global controller reserves $X_c^A \in \mathbb{Z}_{\geq 0}$ subcarriers for the communication of the mobile devices in the coverage of each AP, where X_c^A is an optimization variable. The total amount of spectrum resources reserved for the communication in the coverage of the L APs is thus $Z_c^A = L \cdot X_c^A \cdot B_{c,0}$, where $B_{c,0}$ is the bandwidth of each subcarrier in Hertz. We consider the scenario where there exists at least one mobile device in the coverage of each AP. In

addition, we consider homogeneity among different mobile devices and APs². We focus on one representative mobile device and one representative AP when deriving the service demands and capacities, and refer to them as the mobile device and the AP, respectively. The number of mobile devices in the coverage of the AP is denoted by N^I . At the beginning of each time slot, with an active probability $\rho_c \in (0, 1]$, which is an optimization variable, each mobile device in the coverage of each AP is scheduled by the local controller at the AP for communication. The number of all mobile devices that are scheduled for communication in the time slot is denoted by N_c^I . Conditioning on that $N^I = n$ and the mobile device is scheduled for communication, the probability mass function (PMF) of N_c^I is

$$\mathbb{P}\{N_c^I = n_c | N^I = n\} = \binom{n_c - 1}{n - 1} \rho_c^{n_c - 1} (1 - \rho_c)^{n - n_c}, \quad (3.1)$$

where $n_c = 1, 2, 3, \dots, n$, and $n = 1, 2, \dots$. As a result, we have $\mathbb{E}[N_c^I | N^I] = 1 + (N^I - 1)\rho_c$. The subcarriers are evenly allocated to the mobile devices scheduled for communication, and the number of the subcarriers allocated to the mobile device is $X_c^I = X_c^A / N_c^I$.

Proposition 1. *Denote the transmission rate over each subcarrier (in bits per time slot) by R_0 and the transmission of the mobile device by R . A lower bound of the expectation of transmission rate R is*

$$\bar{R} = \frac{R_0 X_c^A}{\mathbb{E}[N^I] + 1/\rho_c - 1}. \quad (3.2)$$

Proof. Please refer to Appendix A. □

The lower bound in (3.2), i.e., $\bar{R}$, corresponds to the communication service capacity. The average amount of data that each mobile device needs to upload per time slot, denoted by $\hat{R}$, corresponds to the communication service demand. With a lower active probability for communication ρ_c , i.e., a higher active probability for sensing, the lower bound of the transmission rate $\bar{R}$ decreases. The reason is that in such cases, the utilization efficiency of the reserved spectrum resources for communication would be reduced. This reflects a potential trade-off between the capacity of the communication service and that of the sensing service, which should be optimized in the network planning.

²We consider homogeneous mobile devices in industrial scenarios, all with identical capabilities and requirements for executing the same type of tasks. In the scenarios with heterogeneous mobile devices, we can first cluster the mobile devices with similar capabilities and requirements. Then, mobile devices in each cluster can be approximated as homogeneous, and the designs in this paper can be applied.

3.2.4 Sensing Model

We consider two kinds of sensing tasks. The first kind is the *target detection task*. Particularly, in each sensing interval, the AP identifies the targets that have newly arrived or become active within the RoIs of the mobile devices in its coverage. At the end of each sensing interval, based on the detection results, the local controller at the AP updates a database that catalogs the targets within the RoIs of the mobile devices in its coverage. The second kind is the *target tracking task*. In particular, within a sensing interval, a target from the database is assigned to either a mobile device or the AP for monitoring across multiple time slots. At the end of each of the time slots, a sensing data sample is collected for the target. At the end of the interval, the accumulated sensing data samples for each target are processed to determine the state of the target. The tracking results are recorded in the target database, providing prior information to facilitate target tracking in subsequent sensing intervals.

Target Detection and Tracking by APs

The AP performs digital beamforming to simultaneously form multiple beams. To avoid the inter-beam interference, different beams are allocated with orthogonal sensing spectrum bands [28, 23], each having the bandwidth $B_{s,0}$. For the AP, a constant number of $\hat{X}_s^A$ spectrum bands are reserved to form $\hat{X}_s^A$ search beams for identifying the targets that have newly arrived or become active within the RoIs of the mobile devices in its coverage [24]. In addition, X_s^A spectrum bands are reserved for each AP to form $X_s^A \in \mathbb{Z}_{\geq 0}$ track beams, where X_s^A is an optimization variable. Each reserved spectrum band can be used to track one associated target [23, 24]. The total amount of spectrum resources reserved for target detection and tracking by the L APs, denoted by Z_s^A , is $Z_s^A = L \cdot (\hat{X}_s^A + X_s^A) \cdot B_{s,0}$.

Target Tracking by Mobile Devices

For tracking the targets by the mobile devices in the coverage of the L APs, the global controller reserves $X_s^I \in \mathbb{Z}_{\geq 0}$ spectrum bands, each having bandwidth $B_{s,0}$, where X_s^I is an optimization variable. As a result, the total amount of spectrum resources reserved for target tracking by all the mobile devices is $Z_s^I = X_s^I \cdot B_{s,0}$. At the beginning of a time slot, a mobile device will enter the sensing mode if not scheduled by the local controller for communication, the probability of which is $1 - \rho_c$. As shown in Fig. 3.4, similar to [123], in each time slot, the mobile device in the sensing mode randomly chooses one of the X_s^I

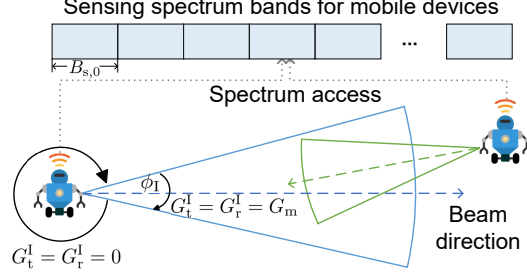


Fig. 3.4: The considered beam pattern of mobile devices and an illustration of the scenario where two mobile devices in the sensing mode interfere each other.

reserved spectrum bands with equal probability $1/X_s^I$, and forms a track beam towards one assigned target [27].

As shown in Fig. 3.4, we consider the following beam pattern of mobile devices in the sensing mode. The width of the beam is ϕ_I , and the antenna gain is $G_m = 2\pi G_0/\phi_I$ in the beam, and zero elsewhere [124]. The power of the signal reflected from a target received at the mobile device, denoted by P_e , can be calculated by $P_e = P_s G_t^I G_r^I c^2 \bar{\sigma} / (4\pi)^3 f_s^2 D^4$ [55], where G_t^I and G_r^I both equal G_m ; P_s represents the transmit power of sensing signals; c represents the speed of light; $\bar{\sigma}$ represents the average value of radar cross section (RCS) of the target; f_s represents the centre frequency of the accessed spectrum band; D represents the distance between the mobile device and the target. The mobile device in the sensing mode will receive interference from any other mobile device that is also in the sensing mode in the current time slot, accessing the same spectrum band for sensing, and residing within each other's beams. For tractability, we consider only the strongest interfering signal to the mobile device, which is the signal from the nearest mobile device. The power of the interference signal is denoted by I_s , and the signal-to-interference ratio (SIR) is denoted by γ_s and calculated as $\gamma_s = P_e/I_s$.

Sensing Data Processing

At the end of each sensing interval, the accumulated sensing data samples for each target in this interval are processed to update the target's state. The total computational overhead for processing the sensing data of one target is denoted by C_0 in CPU cycles. The constant computational capability of each mobile device in CPU cycles per second is denoted by F_e^I . For processing the sensing data collected by each AP, the amount of reserved edge computing resources at the AP is denoted by F_e^A , which is an optimization variable. The total amount of edge computing resources reserved for the L APs is $Z_e^A = L \cdot F_e^A$. The

delay in processing the sensing data collected by the mobile device and the AP within a sensing interval, denoted by T_e^I and T_e^A , respectively, is calculated as $T_e^I = N_I^U C_0 / F_e^I$, and $T_e^A = N_A^U C_0 / F_e^A$. Here, N_A^U and N_I^U represent the numbers of targets that are assigned to the AP and the mobile device at the beginning of a sensing interval.

Sensing Requirements

Multi-dimensional requirements, including the quality, volume, and processing time of sensing data, are considered for the sensing service.

Firstly, the sensing data should have a high quality to provide accurate information of the targets. We use the SIR to assess the data quality, since it has been shown to significantly influence the accuracy of target state estimation [8] and target activity recognition [48]. Consider the randomness in the SIR across different sensing data samples, we adopt the constraint in a probabilistic form [56]: $\mathbb{P}\{\gamma_s \geq \hat{\gamma}_s\} \geq \hat{P}$, where $\hat{\gamma}_s$ is the SIR threshold, and $\hat{P}$ is the required minimum probability that the actual SIR meets or exceeds the SIR threshold.

With an increasing distance between a mobile device and a target, the power of the received sensing signal from the target decreases, which leads to a higher chance of violating the SIR requirement. To satisfy the SIR requirement, the radius of the on-device sensing subregion needs to be properly determined, and the expected number of targets that can be assigned with the mobile device should satisfy the constraint:

$$\mathbb{E}[N_I^U] \leq \mathbb{P}\{D^{I,U} \leq D_{\max}^{I,U}\} \cdot \mathbb{E}[N_{I,1}^U], \quad (3.3)$$

where $D^{I,U}$ represents the distance between a target to its closest mobile device; $\mathbb{P}\{D^{I,U} \leq D_{\max}^{I,U}\}$ is the probability that the distance is smaller than the radius of the on-device sensing subregion; $\mathbb{E}[N_{I,1}^U]$ is the expected number of targets that are closer to the mobile device than to any other mobile devices.

Secondly, the sensing data should have a large volume to provide sufficient information of the targets. We require that the average number of times each target is tracked per time slot should be higher than a threshold $\hat{\rho}_s \in (0, 1]$ [26]. Note that in each time slot, the expected number of tracking attempts that can be conducted by the mobile device and by the AP is $1 - \rho_c$ and X_s^A times, respectively. As a result, the sensing data volume requirement constrains the number of targets that can be assigned to the mobile device and the AP, as follows:

$$\mathbb{E}[N_I^U] \leq \frac{1 - \rho_c}{\hat{\rho}_s}; \quad \mathbb{E}[N_A^U] \leq \frac{X_s^A}{\hat{\rho}_s}. \quad (3.4)$$

Thirdly, the processing time of the sensing data should be low to ensure the freshness of sensing results. We require that the expected time for the mobile device and the AP to process the sensing data collected in each interval, i.e., $\mathbb{E}[T_e^I]$ and $\mathbb{E}[T_e^A]$, should be less than the duration of a sensing interval, or equivalently, T time slots:

$$\frac{\mathbb{E}[N_I^U] C_0}{F_e^I} \leq \tau T; \quad \frac{\mathbb{E}[N_A^U] C_0}{F_e^A} \leq \tau T. \quad (3.5)$$

Sensing Service Demand and Capacity

The number of the targets in the RoIs in the coverage of the AP is denoted by N^U . The expectation of N^U corresponds to the sensing service demand in the coverage of the AP. The demand is calculated by multiplying the expected number of mobile devices with that of the targets within the RoI for each mobile device:

$$\mathbb{E}[N^U] = \mathbb{E}[N^I] \cdot \left(\mathbb{P}\{D^{I,U} \leq \hat{D}\} \cdot \mathbb{E}[N_{I,1}^U] \right), \quad (3.6)$$

where $\mathbb{P}\{D^{I,U} \leq \hat{D}\}$ is the probability that the distance between a target to the closest mobile device is smaller than the radius of the RoI of the mobile device.

Combining the constraint on sensing data volume requirement in (3.4), the constraint on sensing data processing time in (3.5), and the constraint due to the maximum number of targets in the on-device sensing subregion in (3.3), we derive the maximum values of $\mathbb{E}[N_I^U]$ and $\mathbb{E}[N_A^U]$ as

$$\overline{N}_I^U \triangleq \min \left\{ \mathbb{P}\{D^{I,U} \leq D_{\max}^{I,U}\} \cdot \mathbb{E}[N_{I,1}^U], \frac{1 - \rho_c}{\hat{\rho}_s}, \frac{\tau T \cdot F_e^I}{C_0} \right\}, \quad (3.7a)$$

$$\overline{N}_A^U \triangleq \min \left\{ \frac{X_s^A}{\hat{\rho}_s}, \frac{\tau T \cdot F_e^A}{C_0} \right\}, \quad (3.7b)$$

which correspond to the sensing service capacity of each mobile device and each AP, respectively.

3.2.5 Problem Formulation

Our objective is to satisfy the demands for the SAC services in the upcoming planning window with the minimal resource consumption. To achieve this objective, a decision

$\Phi = \{D_{\max}^{\text{I,U}}, \rho_c, X_c^{\text{A}}, X_s^{\text{I}}, X_s^{\text{A}}, F_e^{\text{A}}\}$ is made, which consists of the radius of the on-device sensing subregion, i.e., $D_{\max}^{\text{I,U}}$, the active probability for communication of each mobile device, i.e., ρ_c , the amount of reserved spectrum resources, i.e., X_c^{A} , X_s^{A} , and X_s^{I} , and the amount of reserved edge computing resources, i.e., F_e^{A} . An optimization problem is formulated as follows:

$$\text{(P1) : } \min_{\Phi} \quad Z = \omega (Z_c^{\text{A}} + Z_s^{\text{I}} + Z_s^{\text{A}}) + \xi Z_e^{\text{A}} \quad (3.8a)$$

$$\text{s.t.} \quad 0 < \rho_c \leq 1, \quad (3.8b)$$

$$X_c^{\text{A}}, X_s^{\text{I}}, X_s^{\text{A}} \in \mathbb{Z}_{\geq 0}, \quad (3.8c)$$

$$F_e^{\text{A}}, D_{\max}^{\text{I,U}} \in \mathbb{R}_{\geq 0}, \quad (3.8d)$$

$$\bar{R} \geq \hat{R}, \quad (3.8e)$$

$$\bar{N}_A^{\text{U}} + \mathbb{E}[N^{\text{I}}] \cdot \bar{N}_I^{\text{U}} \geq \mathbb{E}[N^{\text{U}}], \quad (3.8f)$$

$$\mathbb{P}\{\gamma_s \geq \hat{\gamma}_s\} \geq \hat{P}, \quad (3.8g)$$

where Z is the overall resource consumption; ω and ξ are the cost of reserving each unit of spectrum resources and edge computing resources, respectively; constraint (3.8e) ensures that the expected transmission rate of each mobile device is greater than a threshold $\hat{R}$; constraint (3.8f) ensures that the expected number of targets that can be tracked with the requirements on the sensing data quantity and processing time satisfied is greater than the expected number of the targets in all the RoIs in the coverage of the AP; (3.8g) ensures that the requirement on sensing data quality evaluated by SIR is satisfied.

Solving Problem (P1) for network planning requires the spatial modeling of mobile devices and targets. In particular, according to (3.2), calculating the communication service capacity, i.e., $\bar{R}$, requires the expected number of mobile devices in the coverage of each AP, i.e., $\mathbb{E}[N^{\text{I}}]$; According to (3.7a), the sensing capacity of each mobile device, i.e., $\bar{N}_I^{\text{U}}$, depends on the spatial distributions of targets; According to (3.6), the sensing service demand in the coverage of each AP, i.e., $\bar{N}_A^{\text{U}}$, depends on the spatial distributions of the mobile devices and the targets, respectively; Moreover, the sensing interference experienced by each mobile device, impacting γ_s in (3.8g), is affected by the spatial distribution of the mobile devices. However, obtaining an accurate spatial model of the mobile devices and targets is challenging. This is because their spatial distributions can be non-stationary. For example, the safe separation distances between robots and human workers may change dynamically based on factors such as human and robot velocities and specific task requirements [125]. Such non-stationarity can result in the drift of a spatial model for the spatial distributions and degrade the effectiveness of a network planning decision based on the

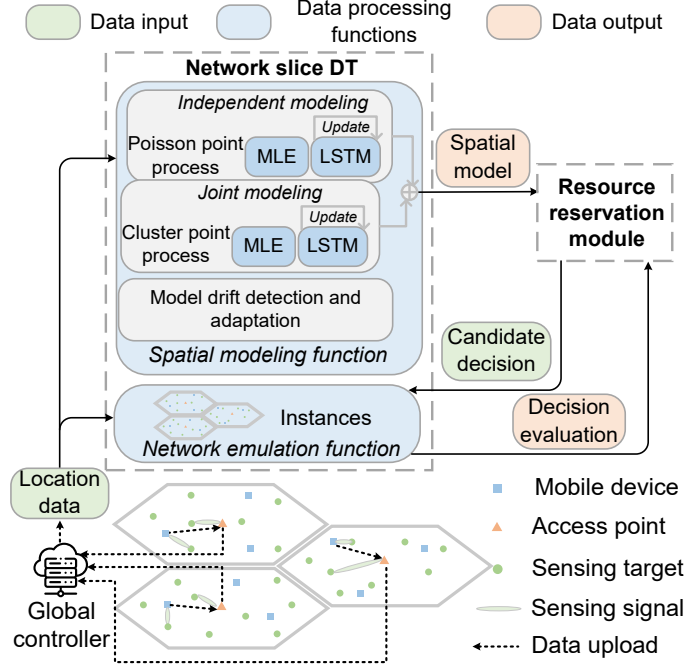


Fig. 3.5: The DT of the sensing slice.

spatial model.

3.3 Network Slice DTs for Spatial Modeling and Network Planning

The DT of a network slice, hereafter referred to as the *slice DT*, is a digital representation of a network slice that is established and updated by collecting and processing data from the network slice for the service-specific prediction and control [38]. We establish network slice DTs for adaptive spatial modeling and efficient and reliable network planning in cooperative ISAC networks. Particularly, in the slice DT corresponding to the sensing service, the spatial distributions of the mobile devices and targets are modeled, while in the slice DT corresponding to the communication service, only the spatial distribution of the mobile devices is modeled. Note that we do not construct digital twins (DTs) for individual mobile devices and targets in spatial modeling, since it may incur significant overhead due to the potentially large number of entities and result in high network planning

complexity.

In Fig. 3.5, we show the main components in the slice DT corresponding to the sensing service. In this section, we first introduce the collection of the spatial distribution data. Then, we introduce a statistical spatial modeling function in the slice DTs based on two statistical spatial models with different modeling granularity. The spatial modeling function detects and adapts to the drift of individual statistical spatial models. Moreover, to ensure that a network planning decision can efficiently satisfy service demands, we propose an network emulation function in the slice DTs to efficiently evaluate a network planning decision.

3.3.1 Spatial Distribution Data Collection

To establish the slice DTs, each AP collects the spatial distribution data of the mobile devices and targets, and uploads them to the global controller every M_0 sensing intervals. Specifically, the locations of the targets assigned to the AP for tracking in this interval are obtained by processing the reflected sensing signals. In addition, the locations of the mobile devices and the targets assigned to the mobile devices are obtained and uploaded by the mobile devices to the AP. The spatial distribution data are used as the input for the slice DTs.

3.3.2 Statistical Spatial Modeling Function

At the end of the k -th planning window, the collected spatial distribution data of mobile devices and targets are processed for the statistical spatial modeling and network planning in the $(k + 1)$ -th planning window. To cope with the potential non-stationarity of the spatial distributions across planning windows, we adopt temporally non-stationary point processes [126]. Specifically, the model parameters are constant in each planning window but can change across planning windows. To tackle the drift in predicting the model parameters, we synergize two statistical models with different numbers of model parameters.

Joint Spatial Modeling

In a joint spatial model, the spatial distributions of mobile devices and targets in the k -th planning window are jointly modeled by a Thomas cluster process. Specifically, the spatial distribution of mobile devices is characterized by a homogeneous Poisson point process

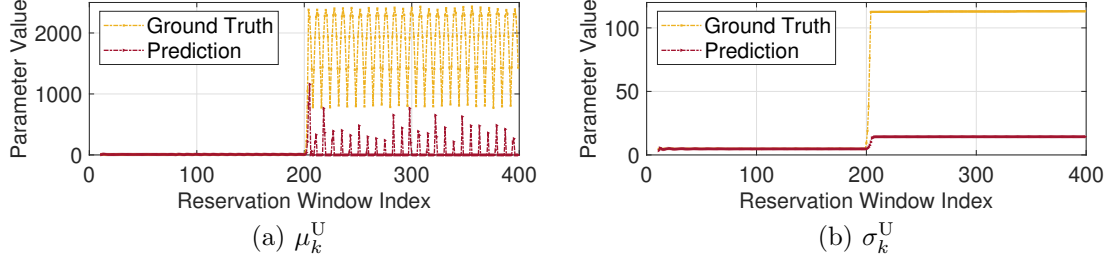


Fig. 3.6: Values of the parameters μ_k^U and σ_k^U in the joint spatial modeling across planning windows.

(PPP) with intensity λ_k^I . The relative two-dimensional locations of the targets clustering around each mobile device follows two identical and independent normal distributions with zero mean and standard deviation σ_k^U [127]. In addition, the number of targets in the cluster of each mobile device follows a Poisson distribution with mean μ_k^U . At the end of the k -th planning window, using the spatial distribution data collected in this planning window, the reference values of the model parameters, i.e., λ_k^I , σ_k^U , and μ_k^U , are determined based on the maximum likelihood estimation (MLE). The prediction values of the three parameters for network planning in the $(k+1)$ -th planning window, denoted by $\tilde{\lambda}_{k+1}^I$, $\tilde{\sigma}_{k+1}^U$, and $\tilde{\mu}_{k+1}^U$, are obtained by the long short-term memory (LSTM) neural networks based on the reference values of the parameters in the previous K_0 windows.

The joint spatial modeling characterizes the spatial correlation between mobile devices and targets with the model parameters μ_k^U and σ_k^U . If the parameters can be accurately predicted for the upcoming planning window, the joint spatial modeling would be accurate. However, when the pattern of their spatial correlation changes, e.g., from attraction to repulsion, the values of the parameters that best fit the actual spatial distributions may exhibit a drastic change. In such a case, the accuracy of parameter prediction and that of the joint spatial modeling can degrade, referred to as *model drift*. To illustrate this, in Fig. 3.6, we show the reference and prediction values of the model parameters μ_k^U and σ_k^U in the joint spatial modeling, which are derived by MLE and the LSTM neural networks, respectively. The simulation settings are detailed in Section 3.5. It can be observed that after the 201-st planning window, the reference values of μ_k^U and σ_k^U exhibit a drastic increase, and the prediction values show an increasing gap with the reference values, indicating model drift.

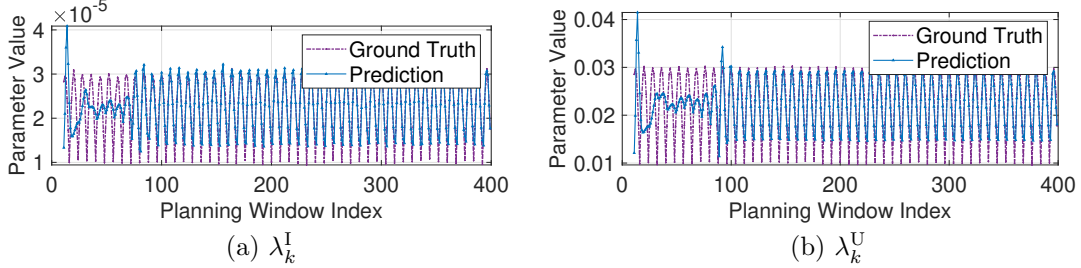


Fig. 3.7: Values of the parameters λ_k^I and λ_k^U in the independent spatial modeling across planning windows.

Independent Spatial Modeling

To cope with the model drift, a spatial model robust to change in the spatial correlation of the mobile devices and targets is needed. We introduce the independent spatial modeling, which independently characterizes the spatial distributions of the mobile devices and targets. In particular, the locations of the mobile devices and targets in the k -th planning window are modeled by two *independent* homogeneous PPPs with intensities λ_k^I and λ_k^U , respectively. Similar to the joint spatial modeling, using the collected data in the k -th planning window, the reference values of the intensity parameters in the two homogeneous PPPs, i.e., λ_k^I and λ_k^U , are determined based on the MLE. The prediction values of the two parameters for the $(k + 1)$ -th planning window, i.e., $\check{\lambda}_{k+1}^I$, $\check{\lambda}_{k+1}^U$, are obtained by LSTM networks based on reference values of the parameters in the previous K_0 windows.

In Fig. 3.7, for the same spatial distribution data as the joint spatial modeling in Fig. 3.6, we show the reference and prediction values of all the parameters in the independent spatial modeling, i.e., λ_k^I and λ_k^U . Note that the reference values evolve over planning windows in a manner similar to the absolute value of a cosine function, which makes the prediction at the lowest points of the reference values challenging and results in large discrepancies at those points. Nevertheless, it can be observed that there is no drastic parameter value change and parameter prediction accuracy degradation since the 201-st planning window. This observation suggests that the independent spatial modeling is more robust to model drift than the joint spatial modeling. However, it is important to note that in scenarios without model drift, the independent spatial modeling would yield lower accuracy than the joint spatial modeling due to its inability to characterize the spatial correlation between mobile devices and targets.

Model Drift Detection

This function monitors the prediction errors in the parameters of the spatial models to detect model drift. We develop a prediction error-based approach for model drift detection [120]. Specifically, the errors in predicting $\{\lambda_k^I, \lambda_k^U, \mu_k^U, \sigma_k^U\}$ are evaluated by the mean absolute percentage error (MAPE). Let H_k^S and $H_k^D \in \{0, 1\}$ indicate whether the independent spatial modeling and the joint spatial modeling, respectively, experience model drift. At the end of the k -th planning window, the following three cases are considered for model drift detection. First, if the prediction of the parameter λ_k^I , which is shared by both spatial modelings, exhibits drastic accuracy degradation, we set $H_k^S = 1$ and $H_k^D = 1$. Second, if the prediction of the parameter λ_k^U , which is unique to the independent spatial modeling, exhibits drastic accuracy degradation, we set $H_k^S = 1$. Third, if parameters μ_k^U and σ_k^U , which are unique to the joint spatial modeling, exhibit drastic accuracy degradation, we set $H_k^D = 1$.

Model Drift Adaptation

In this function, the following two approaches are designed to adapt to any model drift that is detected: (i) *Model Update*: when model drift is detected, LSTM neural networks for predicting the parameters corresponding to the model drift are updated. In particular, in the planning windows after the detection, the reference values of the parameters will be collected and used to retrain the neural networks; (ii) *Model Ensemble*: when model drift is detected, the ensemble learning technique is exploited, and an ensemble spatial model that incorporates both the independent and the joint spatial models, is output by the model drift adaptation function. In the absence of model drift, only the joint spatial model is output, since it offers potentially higher accuracy compared to the independent model and lower complexity for network planning compared to the ensemble model.

3.3.3 Network Emulation Function

It is infeasible to directly fuse the joint and independent spatial models in the ensemble spatial model for network planning. Instead, we can evaluate and select from the two *candidate* network planning decisions that are respectively obtained using the two spatial models. For the evaluation, we introduce a network emulation function that consists of the following two steps: (i) *Instance Construction*: multiple instances of both the SAC slices are created. Each instance of a communication slice involves deterministically distributed APs and mobile devices, while each instance of a sensing slice involves deterministically

distributed APs, mobile devices, and targets. For example, the locations of all the mobile devices and targets in the RoIs of the mobile devices at the end of the M_0 -th sensing interval in the k -th planning window, along with the locations of the APs, are used to create one instance of the sensing slice; (ii) *Decision Evaluation*: a candidate network planning decision is implemented on each of the instances. Then, the service demand and capacity of each instance are measured to evaluate the corresponding candidate decision. The measurement approach and the evaluation metric for a network planning decision are described in Section 3.4.2.

3.4 DT-Based Network Planning Algorithm

In this section, based on the statistical spatial models and the network emulation functions in the slice DTs, we propose a network planning algorithm.

3.4.1 Statistical Spatial Modeling-Based Decision Making

In this subsection, we first derive the formulas for estimating the service demands and capacities in Problem (P1). Then, we analyze Problem (P1) for transforming the problem and developing an efficient algorithm to obtain a candidate network planning decision for the $(k + 1)$ -th planning window. For brevity, we omit the subscript $k + 1$ in the notation of the parameters that are predicted by LSTM neural networks.

Estimation of the Communication Service Capacity

In both independent and joint spatial models, the mobile devices are modeled by a homogeneous PPP with the intensity $\check{\lambda}^I$. Given that at least one mobile device exists, the expected number of mobile devices in the coverage of the AP with the area A_0 is given by $\mathbb{E}[N^I] = \check{\lambda}^I A_0 / (1 - e^{-\check{\lambda}^I A_0})$. By substituting $\mathbb{E}[N^I]$ in (3.2) with this expression, the communication service capacity $\bar{R}$ in (3.2) can be calculated as

$$\bar{R} = \frac{\rho_c R_0 X_c^A}{1 + \left(\frac{\check{\lambda}^I A_0}{(1 - e^{-\check{\lambda}^I A_0})} - 1 \right) \rho_c}. \quad (3.9)$$

Estimation of the Sensing Service Demand and Capacity

According to (3.6), estimating the sensing demand in the coverage of each AP, i.e., $\mathbb{E}[N^U]$, requires (i) the probability that a target is located in the RoI of the nearest mobile device, i.e., $\mathbb{P}\{D^{I,U} \leq \hat{D}\}$, and (ii) the expected number of targets closest to the mobile device, i.e., $\mathbb{E}[N_{I,1}^U]$.

First, the cumulative distribution function of the distance between a target to the closest mobile device, i.e., $D^{I,U}$, is denoted by $\mathbb{P}\{D^{I,U} \leq d^{I,U}\}$ and approximated as follows. Given the independent spatial model, according to the contact distance distribution of a homogeneous PPP in [128], we have $\mathbb{P}\{D^{I,U} \leq d^{I,U}\} = 1 - \exp\left[-(d^{I,U})^2 \pi \check{\lambda}^I\right]$. Given the joint spatial model, according to the distribution of the distance in any cluster of a Thomas cluster process in [127], we have $\mathbb{P}\{D^{I,U} \leq d^{I,U}\} = 1 - \exp\left[-(d^{I,U})^2 / (2(\check{\sigma}^U)^2)\right]$. Accordingly, $\mathbb{P}\{D^{I,U} \leq \hat{D}\}$ can be calculated.

Second, the expected number of targets closest to the mobile device, i.e., $\mathbb{E}[N_{I,1}^U]$, is approximated as follows. Given the independent spatial model, $\mathbb{E}[N_{I,1}^U]$ is approximated as $\mathbb{E}[N_{I,1}^U] = \check{\lambda}^U / \check{\lambda}^I$, which is the ratio of the intensity of the targets to that of the mobile devices, due to the identical randomness across different mobile devices. Given the joint spatial model, $\mathbb{E}[N_{I,1}^U]$ is approximated as $\mathbb{E}[N_{I,1}^U] = \check{\mu}^U$, which is the expected number of targets in the cluster formed by the mobile device. According to (3.7a), estimating the sensing capacity of each mobile device, i.e., $\overline{N}_I^U$, requires $\mathbb{P}\{D^{I,U} \leq D_{\max}^{I,U}\}$ and $\mathbb{E}[N_{I,1}^U]$, which have been derived. The sensing capacity of each AP, i.e., $\overline{N}_A^U$, is not affected by the spatial model, and has been derived in (3.7b).

Problem Analysis

We analyze the properties of Problem (P1) as follows.

Proposition 2. *In at least one optimal solution of Problem (P1), the radius of the on-device sensing subregion, i.e., $D_{\max}^{I,U}$, is given by*

$$D_{\max}^{I,U} = \left(\frac{\bar{\sigma}}{\hat{\gamma}_s} \cdot (-\ln \hat{P}) \cdot \frac{1}{\phi_1^2 \check{\lambda}^I} \right)^{\frac{1}{4}} \cdot \left(\frac{X_s^I}{1 - \rho_c} \right)^{\frac{1}{4}}, \quad (3.10)$$

which is also the maximum distance between a target and an mobile device for satisfying the sensing SIR requirement in (3.8g).

Proof. Please refer to Appendix B. □

Based on Proposition 2, we can transform constraint (3.8g) equivalently using (3.10). To make Problem (P1) more tractable, we relax the integer decision variables, i.e., X_s^I, X_c^A, X_s^A , to real-valued variables. Then, to efficiently solve the problem, we derive necessary conditions for optimality as follows.

Proposition 3. Proposition 3: *The optimal solution of Problem (P1) must satisfy the following three conditions:*

$$\frac{1 - \rho_c}{\widehat{\rho}_s} \leq \min \left\{ \mathbb{E} [N_{I,1}^U], \frac{\tau T \cdot F_e^I}{C_0} \right\}; \quad (3.11a)$$

$$\frac{1 - \rho_c}{\widehat{\rho}_s} = \mathbb{P} \{ D^{I,U} \leq D_{\max}^{I,U} \} \cdot \mathbb{E} [N_{I,1}^U]; \quad (3.11b)$$

$$\frac{X_s^A}{\widehat{\rho}_s} = \frac{\tau T \cdot F_e^A}{C_0}. \quad (3.11c)$$

Proof. Please refer to Appendix C. □

Condition (3.11a) specifies a minimum value of ρ_c since the right-hand side is independent of any decision variable in Problem (P1); (3.11b) gives the relation between ρ_c and X_s^I since $\mathbb{P} \{ D^{I,U} \leq D_{\max}^{I,U} \}$ is a function of only these two decision variables; (3.11c) gives the relation between X_s^A and F_e^A .

Problem Transformation and Solution

Based on **Proposition 2** and **Proposition 3**, given the value of the decision variable ρ_c , the optimal values of the other four decision variables can be determined in closed forms. First, the minimum spectrum reservation for communication, X_c^A , can be obtained by converting the inequality constraint (3.8e) to an equality constraint. Second, the optimal amount of spectrum resources reserved for sensing by mobile devices, X_s^I , can be found by solving (3.11b). Third, the optimal spectrum reservation for sensing by APs, X_s^A , can be obtained by converting the inequality constraint (3.8f) to an equality constraint. Given the optimal X_s^A , the optimal edge computing network planning, F_e^A , can be obtained based on (3.11c). As a result, the original multi-variable optimization problem can be transformed into a single-variable problem concerning ρ_c . The transformed problem can be solved by a brute-force search. Then, the real values in the decision are rounded up to integers. Given the joint spatial model or the independent spatial model, a candidate network planning decision can be respectively obtained.

3.4.2 Network Emulation-Based Decision Evaluation

Based on the network emulation function of the slice DTs in Section 3.3.3, a candidate network planning decision, denoted by Φ , can be evaluated in the following way.

Evaluation under Communication Service Provision

Evaluating the *average* transmission rate of mobile devices requires network emulation over a long period. To reduce the complexity, we use the *expectation* of the transmission rate derived in (3.9). Given the spatial distribution data of mobile devices in all the constructed instances, the intensity of mobile devices is determined by MLE and used to replace the one predicted by an LSTM network, i.e., $\check{\lambda}^I$ in (3.9). Then, given the value of ρ_c in a network planning decision Φ , the lower bound of the expected achievable transmission rate, i.e., $\bar{R}$, is calculated as the network emulation-based estimation of the communication service capacity. The evaluation metric chosen here is the relative difference between the service demands and capacities. Specifically, for the communication service, the relative difference is denoted by $\Delta_c(\Phi)$ and calculated as: $\Delta_c(\Phi) = |\bar{R} - \hat{R}|/\hat{R}$, which can reflect both the resource over-provision, where the communication capacity exceeds the communication demand, i.e., $\bar{R} > \hat{R}$, and resource under-provision, i.e., $\bar{R} < \hat{R}$.

Evaluation under Sensing Service Provision

Similarly, given the candidate network planning decision Φ , the number of the mobile devices, the number of targets that can be monitored by the mobile devices, and the number of targets that reside in the RoIs of all the mobile devices in the network instances are obtained and averaged as the network emulation-based estimation of $\mathbb{E}[N^I]$, $\bar{N}_I^U$, and $\bar{N}^U$, respectively. The relative difference between the sensing demand and sensing capacity is denoted by $\Delta_s(\Phi)$ and calculated as $\Delta_s(\Phi) = \left| \bar{N}^U - \left(\bar{N}_A^U + \mathbb{E}[N^I] \cdot \bar{N}_I^U \right) \right| / \bar{N}^U$.

Overall Evaluation of a Network Planning Decision

Given a candidate network planning decision Φ , the average difference between the demands and capacities for the SAC services is denoted by $\Delta(\Phi)$ and calculated as $\Delta(\Phi) = (\Delta_c(\Phi) + \Delta_s(\Phi))/2$, which is used for the overall evaluation of the network planning decision.

3.4.3 Overall DT-Based Network Planning Algorithm

Based on the proposed closed-form candidate network planning decision making and the network emulation-based decision evaluation approaches, we propose a network planning algorithm that can adapt to the drift of individual spatial models as follows: (i) *Case 1 (Model Drift Not Detected)*: When model drift is not detected, i.e., $H_k^S = 0$ and $H_k^D = 0$, only the joint spatial model is output from the slice DT. The network planning decision based on the joint spatial model, denoted by Φ^D , is obtained according to Section 3.4.1, and adopted for the $(k+1)$ -th planning window. (ii) *Case 2 (Model Drift Detected)*: When model drift is detected for the joint modeling, i.e., $H_k^S = 1$ or $H_k^D = 1$, the slice DTs would output an ensemble spatial model consisting of both the joint and the independent spatial models. In this case, the two candidate network planning decisions are respectively derived as Φ^S and Φ^D . Then, the two candidate decisions are respectively evaluated by the network emulation function according to Section 3.4.2, and the evaluation results $\Delta(\Phi^S)$ and $\Delta(\Phi^D)$, reflecting the respective discrepancies between service demands and service capacities, are obtained. The candidate decision with a smaller demand-capacity discrepancy is selected as the final network planning decision for the $(k+1)$ -th planning window.

3.5 Simulation Results

In this section, we demonstrate by simulations the performance of our proposed DT-based network planning algorithm for cooperative ISAC networks.

3.5.1 Simulation Settings and Benchmarks

The synthetic spatial distributions of the mobile devices and targets in the simulations are generated in the following way. The locations of mobile devices are generated following a homogeneous PPP, while the locations of targets are generated in two steps. In the first step, the number of targets in the RoI of each mobile device is determined by sampling from a Poisson distribution. In the second step, targets are randomly and uniformly generated within the RoI of each mobile device. A thinning approach is then used to determine whether to retain each target based on a probability, until the predetermined number of targets for each mobile device is reached. Specifically, two types of mobile devices are considered, with devices set to type I and type II with probabilities ν and $1-\nu$, respectively. The probabilities to retain a generated target for type I and type II mobile devices are given by $D^{I,U}/\hat{D}$ and $1 - D^{I,U}/\hat{D}$, respectively, where $D^{I,U}$ is the distance between the target and

Table 3.1: Main simulation settings of the ISAC network

Parameter	Value
Beamwidth of mobile devices for sensing, ϕ_I	$\pi/6$
Bandwidth of a communication subcarrier, $B_{c,0}$	0.015 MHz
Transmit power for sensing, P_s	20 dBm
Bandwidth of a sensing band, $B_{s,0}$	1 MHz
Average radar cross section, $\bar{\sigma}$	0.5 m ²
Side length of each AP's coverage, r_0	500 m
Duration of a time slot, τ	1 ms
Number of time slots in a sensing interval, T	1000
Number of sensing intervals in a planning window, M	500
Location data collection period in sensing intervals, M_0	10
Number of APs in the cluster, L	3
CPU cycles for sensing one target, C_0	1×10^8 cycles
CPU frequency of each mobile device, F_e^I	2 GHz
SIR requirement for sensing, $\hat{\gamma}_s$	20 dB
Threshold for SIR requirement satisfaction, $\hat{P}$	0.95
Communication throughput requirement, $\hat{R}$	1 kb/slot
Sensing frequency requirement for each target, $\hat{\rho}_s$	0.05
Radius of the RoI for each mobile device, $\hat{D}$	10 m
Cost of reserving a unit of spectrum resources, ω	1
Cost of reserving a unit of edge computing resources, ξ	1×10^{-3}

the mobile device, and $\hat{D}$ is the radius of the RoI. The spatial correlation pattern between the mobile devices and the sensing targets around them differs for the two types of mobile devices. When the value of ν changes, the proportions of the two types of the mobile devices and the overall spatial correlation pattern will be changed, which may result in the drift of joint spatial model which characterizes the spatial correlation. The other main simulation settings are given in Table 3.1.

To generate the spatial distribution data across planning windows, the parameters of the homogeneous PPP and the Poisson distribution are changed based on the absolute values of cosine functions with the angular frequency $\pi/8$. Each simulation run includes 400 consecutive planning windows, and for each simulation setting, we conduct 5 simulation runs and evaluate the average performances. The following three performance metrics are adopted: (i) *Actual Service Satisfaction Ratio*: this metric represents the average satisfaction ratios for the sensing and communication services as observed at the end of each planning window. Each service's satisfaction ratio is calculated as the minimum between 1 and the ratio of the service capacity to the service demand; (ii) *Overall Resource Consumption*: this metric represents the overall cost of reserving the spectrum resources and edge computing resources for the SAC services, as defined in (5.8a); (iii) *Modeling*

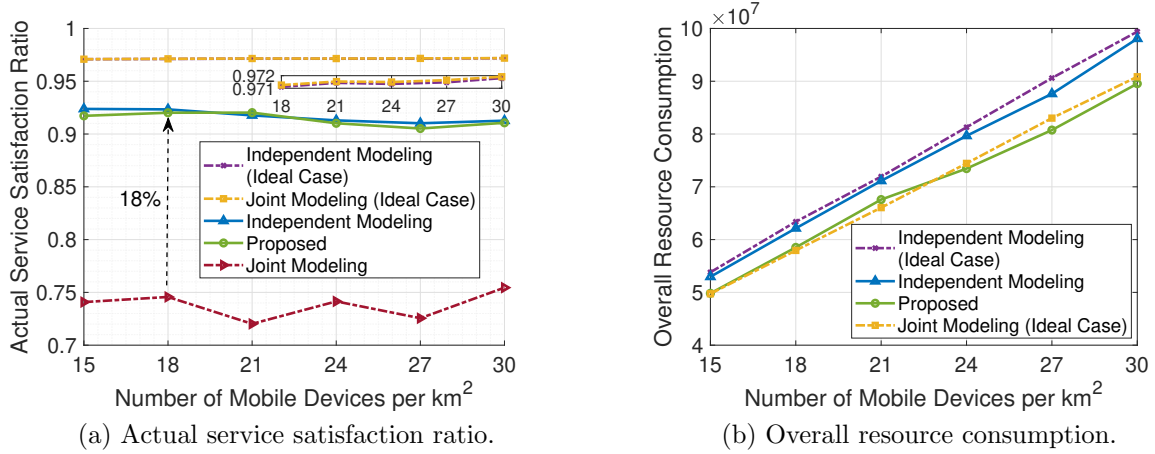


Fig. 3.8: Performance evaluation with varying number of mobile devices per square kilometer.

Error: this metric represents the average prediction error of the parameters in a spatial model. A sliding average approach is used with the width of 20 planning windows.

We compare our proposed algorithm with the following benchmarks: (i) *Independent Modeling (Ideal Case)*: use the independent spatial modeling with perfect parameter prediction; (ii) *Independent Modeling*: use the independent spatial modeling with LSTM neural networks to predict model parameters, while the proposed model drift detection and the model update approaches are not implemented; (iii) *Joint Modeling (Ideal Case)*: use the joint spatial modeling with perfect parameter prediction; (iv) *Joint Modeling*: use the joint spatial modeling with LSTM neural networks to predict model parameters, while the proposed model drift detection and model update approaches are not implemented.

3.5.2 Effectiveness of DT-Based Network Planning Algorithm

In this subsection, we demonstrate the performance of the DT-based network planning algorithm. In each simulation, the value of ν is set to 0.1 and 0.7 before and after the 201-st planning window. The average number of targets in the RoI of each mobile device is set to be 5.5 and that of the mobile devices varies from 15 to 30 per square kilometre. In Fig. 3.8(a), we show the actual service satisfaction ratio, and in Fig. 3.8(b), we show the overall resource consumption.

First, in the two ideal cases where parameter predictions are perfect, we compare the

performances of the joint modeling and independent modeling. It can be observed from Fig. 3.8(a) that the use of a single model in the ideal cases result in similarly high service satisfaction ratios. From Fig. 3.8(b), it can be observed that the use of the joint modeling in the ideal case results in the network planning decisions that consume less resources than the use of independent spatial modeling in the ideal case. This is because, even when the independent modeling may accurately estimate the expected number of targets in the whole RoI of each mobile device, it underestimates the number in the on-device sensing subregion, since the spatial attraction between mobile devices and targets is not characterized. Consequently, the demand for sensing services can be accurately estimated, whereas the sensing capacity of mobile devices is under-estimated. As a result, although the service demand can be satisfied, spectrum and edge computing resources would be over-provisioned.

Second, in the non-ideal cases where model parameters are predicted by LSTM neural networks, we compare the performances of the independent modeling, joint modeling, and our proposed DT-based network planning algorithms. Unlike in the ideal cases where the joint modeling outperforms the independent modeling, in the non-ideal cases, it can be observed from Fig. 3.8(a) that, the joint modeling for network planning results in a much lower actual service satisfaction ratio than the independent modeling. This is because, as shown in Fig. 3.6, after the value of ν changes in the 201-st planning window, the joint spatial modeling exhibits model drift, while the independent spatial modeling does not. Specifically, as shown in Fig. 3.6(a), the average number of targets clustering around each mobile device is under-predicted by the joint modeling, which results in the under-prediction of the sensing service demand. In addition, as shown in Fig. 3.6(b), the standard derivation of the distance between a target and a mobile device is under-predicted by the joint modeling, which results in the over-prediction of mobile devices' sensing capability. In addition, it can be observed from Fig. 3.8(a) that using the DT-based network planning algorithm results in nearly the same service satisfaction ratio as the independent spatial modeling-based network planning. Nevertheless, it can be observed from Fig. 3.8(b) that, the use of the DT-based network planning algorithm results in lower resource consumption compared with the independent spatial modeling-based network planning algorithm.

From the observations in the non-ideal cases, our proposed DT-based network planning algorithm achieves the best performance. In the following subsections, we illustrate the benefits of our proposed model update and model ensemble approaches, respectively, where the average number of targets in the RoI of each mobile device is 5.5 and that of the mobile devices is 15.

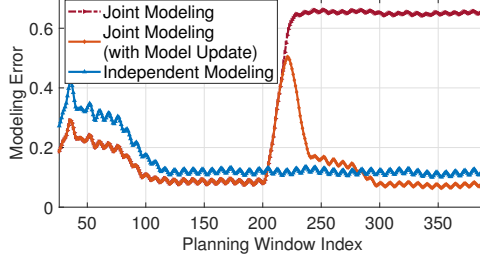


Fig. 3.9: Modeling error across planning windows.

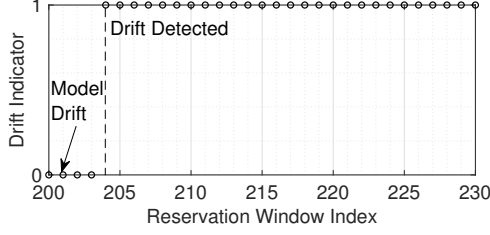
3.5.3 Effectiveness of Model Update

In Fig. 3.9, we show the modeling error across planning windows. From Fig. 3.9, it can be observed that, after the value of ν changes in the 201-st planning window, the modeling error of the joint modeling drastically increases, while that of the independent modeling does not. In contrast, with model drift detection and model update by retraining the LSTM neural networks, the modeling error of the joint modeling quickly falls down after the increase. This observation demonstrates that our model update approach can enable the joint spatial modeling to tackle the model drift issue, making it suitable for network planning in the presence of model drift.

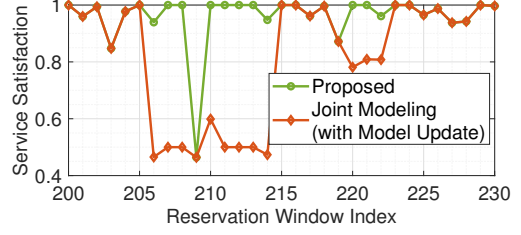
3.5.4 Effectiveness of Model Ensemble

In this subsection, we demonstrate the performance with the model drift of the joint spatial modeling occurring at different frequencies, which are defined as the total numbers of drift occurrences that are uniformly distributed across consecutive 400 planning windows. To simulate an occurrence of the model drift, the value of ν alters between 0.1 and 0.7.

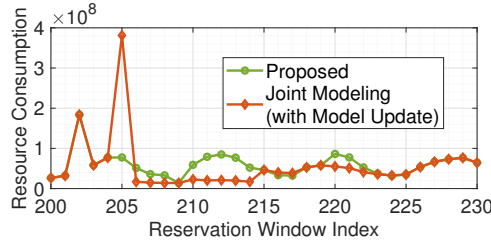
First, in Fig. 3.10, we show the per-window performances from the 200-th to the 230-th planning window, where drift occurs 3 times in 400 windows. In Fig. 3.10(a), we show the model drift detection result. From Fig. 3.10(a), it can be observed that the model drift, occurring in the 201-st planning window, is successfully detected at the 204-th planning window. In Fig. 3.10(b), we show the actual service satisfaction ratio. From the figure, it can be observed that, after the model drift occurs, the actual service satisfaction ratio, by using the joint modeling with the model update, first decreases, and then increases. In addition, it can be observed that by using the model ensemble approach in the DT-based adaptive modeling, a higher service satisfaction ratio can be achieved in multiple planning windows. This is because, by using the model ensemble approach, when model



(a) Model drift indicator H_k^D .



(b) Actual service satisfaction ratio.

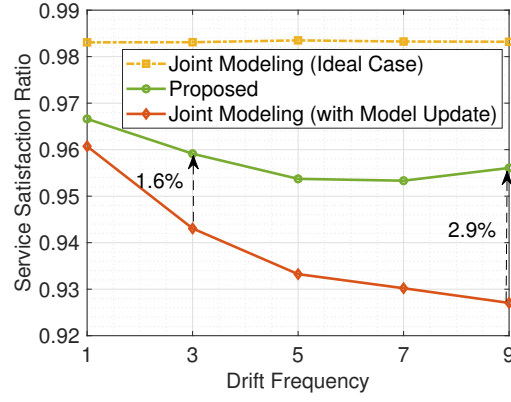


(c) Overall resource consumption.

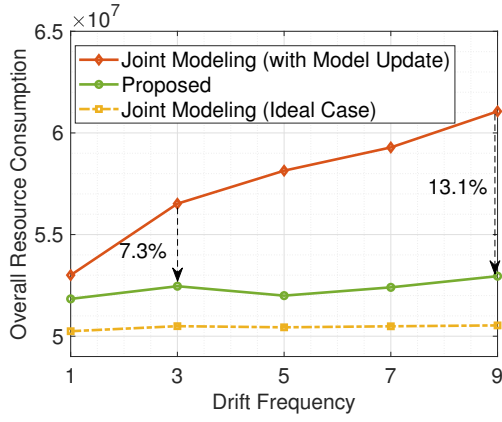
Fig. 3.10: Performance evaluation in the early planning windows after model drift occurs.

drift is detected, both the joint and independent spatial models are output from the slice DTs for network planning, and the network emulation function in the slice DTs evaluates the network planning decisions derived from the two spatial models, and selects the better decision. Similarly, from Fig. 3.10(c), it can be observed that, the model ensemble approach demonstrates its effectiveness by preventing resource over-provision in the 205-th planning windows, while avoiding under-provision in the remaining windows between the 206-th and 214-th.

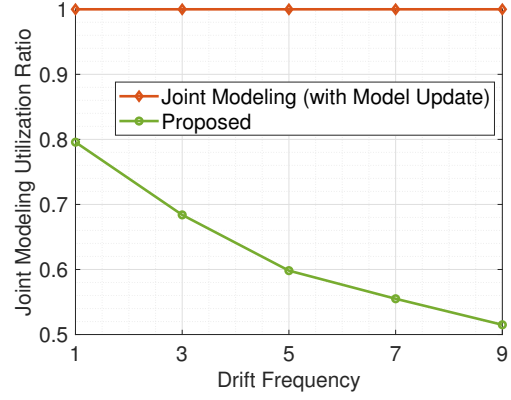
Second, we show the average performance across planning windows when the model drift frequency takes different values. Note that we only consider odd integer values for the drift frequency because they result in equal proportions of the two types of mobile devices. Consequently, although the service provisions differ between the two types, the overall service provision performance in the ideal case remains the same across the different drift frequencies. This consistency helps clearly reveal the impact of the drift frequency on our proposed scheme and the benchmark schemes. In Fig. 3.11(a) and Fig. 3.11(b), we show the actual service satisfaction ratio and the overall resource consumption, respectively. From the two figures, it can be observed that compared with always using the joint modeling, using the model ensemble approach in our proposed DT-based spatial modeling function improves the actual service satisfaction ratio and decreases the overall resource



(a) Actual service satisfaction ratio.



(b) Overall resource consumption.



(c) Joint modeling utilization ratio.

Fig. 3.11: Performance evaluation under varying model drift frequency.

consumption. In addition, with a higher model drift frequency, the advantage of DT-based spatial modeling function, in terms of improving the actual service satisfaction and reducing the resource consumption, increases. This is because, with the increase of drift frequency, the joint modeling would result in degraded performance, i.e., reduced service satisfaction ratio and increased resource consumption. Nevertheless, in our proposed DT-based spatial modeling function, as shown in Fig. 3.11(c), the joint modeling utilization ratio, defined by the percentage of the planning windows when the joint spatial modeling is selected, decreases with the increase of the drift frequency. In this way, the service provision performance becomes less invariant to the increase of drift frequency, similar to ideal case as shown in Fig. 3.11(a) and Fig. 3.11(b).

The observations in this subsection demonstrate that it is not always beneficial to characterize the spatial correlation between mobile devices and targets for resource management, especially when the spatial correlation can experience drastic variations. Instead, by using our proposed DT-based spatial modeling and network emulation functions, the spatial correlation can be properly characterized for effective resource management in co-operative ISAC networks.

3.6 Conclusion

In this paper, we have proposed a novel drift-adaptive slicing-based resource management scheme to support the efficient collaboration of mobile devices and APs in SAC service provision. We have developed DTs of the SAC slices for drift-adaptive spatial modeling and efficient network emulation, which empowers efficient and reliable slicing-based resource management decision-making in non-stationary environments. In addition to network-level spatial modeling and emulation, our proposed DT design can also be applied to adaptively characterize the service demand and satisfaction of individual users.

Chapter 4

Task Offloading for DNN-Based Sensing Data Processing

In this chapter, we propose a novel DT-assisted approach to offloading DNN-based sensing data processing tasks, referred to as DNN tasks. We consider a scenario where, in the local processing of each DNN task, an end device adaptively determines whether and when to offload the DNN task to the edge server, such that on-device and edge computing resources can be efficiently leveraged for optimizing task performance. To enhance the adaptivity, a DT is constructed to evaluate all potential offloading decisions for each DNN task, which provides augmented training data for a machine learning-assisted decision-making algorithm. Then, another DT is constructed to estimate the inference status at the device to avoid frequently fetching the status information from the device, thus reducing the signaling overhead. We also derive necessary conditions for optimal offloading decisions to reduce the offloading decision space. Simulation results demonstrate the outstanding performance of our DT-assisted approach in terms of balancing the tradeoff among inference accuracy, delay, and energy consumption.

4.1 Background and Motivation

In recent years, DNNs, as a representative AI technique, have demonstrated powerful abilities to automatically extract features for pattern recognition, making them advantageous for sensing data processing in IoT applications [3] and empowering the Artificial Intelligence of Things (AIoT) paradigm [129]. To address the substantial computation overhead

and demand for DNN-based sensing tasks, referred to as DNN tasks, a communication-assisted sensing framework under the ISAC paradigm can be beneficial. First, by assigning sensing targets to network infrastructure such as APs, the associated edge servers can be utilized for processing the sensing data collected from the targets. In addition, an end device with limited DNN inference capability, referred to as AIoT device, can upload the raw or processed sensing data to edge servers for completing the processing.

DNN partitioning is a promising solution for achieving adaptive offloading of DNN-based sensing tasks from AIoT devices to edge servers. In DNN partitioning, the inference of a multi-layer DNN can be partitioned into two parts, i.e., on-device inference and edge inference [32]. Specifically, an AIoT device executes the layers before a partition layer, i.e., on-device inference, and uploads the intermediate result, as the input to the partition layer, to an edge server. The edge server then executes the remaining layers of the DNN, i.e., edge inference, to obtain the final inference result. The partition layer for a DNN task determines computing load of on-device inference and edge inference, respectively, and the size of data to be offloaded. Therefore, the partition layer should be properly chosen to minimize the overall delay and energy consumption for task processing [130]. When the edge server is heavily loaded, the overall delay to process a task can be high, even if the optimal partition layer is chosen for the task. In this case, the DNN inference can be early exited by AIoT devices for some tasks instead of offloading them to the edge server for completing the inference[131]. In this way, the overall delay of task processing can be reduced at the cost of reduced inference accuracy [132]. By determining whether to offload a DNN task to the edge server or not, such a tradeoff can be made as needed.

For the multi-layer and multi-exit DNNs, DNN task offloading decisions can be made on whether and when an AIoT device should stop on-device inference and upload intermediate results to an edge server to complete the inference. Such decision-making should adapt to the computing workload status at the device and the edge server, which is challenging since the workload status can change dynamically due to stochastic task generation at each AIoT device covered by the edge server [133]. In particular, conventional approaches that make the offloading decision for each DNN task only upon its generation is unlikely to be optimal [33]. This is because on-device inference delay for a task can be as long as hundreds of milliseconds for executing one convolutional layer [132], which can result in *i*) a drastic change of the instantaneous workload status between the generation of the task and the completion of the on-device inference for the task, and *ii*) difficulty in predicting such a change especially when the dynamics of task arrival is unavailable [132, 91].

The DT paradigm, proposed as a promising next stage of network virtualization [38], can be leveraged to facilitate adaptive device-edge collaboration on DNN inference. Through data synchronization, DTs can be constructed as digital representations of physical objects

or processes to manage network data [134], estimate network status (e.g., the running states of IoT devices [135]), and predict network performance (e.g., the end-to-end delay of network slices[116]). In addition, DTs can empower network emulations for efficient network management [38], e.g., constructing a simulation environment for pre-training a machine learning-assisted task offloading algorithm [136]. For the case of device-edge collaborative DNN inference, DTs can empower machine learning-assisted offloading decision-making by providing augmented training data. For example, after a DNN task is offloaded to an edge server, DTs can be used to evaluate alternative offloading decisions of the task in hypothetical scenarios, i.e., offloading the task later or locally completing the task.

In this chapter, we leverage DTs to facilitate adaptive offloading of DNN-based sensing data processing tasks. We consider a scenario where a full-size DNN is deployed at an edge server for high-accuracy inference, and a shallow DNN, which consists of the first several layers of the full-size DNN concatenated by an exit branch, is deployed at an AIoT device for low-latency on-device inference. In addition, the on-device and edge server computing workloads are dynamic due to stochastic task arrival. The objective is to optimize the performance of DNN tasks in terms of balancing inference accuracy, delay, and energy consumption by determining the offloading of DNN tasks generated by the AIoT device. To achieve this objective, we propose a DT-assisted approach to offloading DNN-based sensing tasks, which determines whether to offload a DNN task each time when the on-device inference status changes, i.e., when a layer of the shallow DNN is about to be locally executed for the task. An offloading decision-making algorithm based on the optimal stopping theorem and assisted by machine learning is developed. Moreover, DTs are constructed in the proposed approach to play the following two roles:

1. Evaluating all potential offloading decisions of each DNN task to provide augmented training data for the learning-assisted offloading decision-making algorithm.
2. Estimating the inference status at the device to avoid frequently fetching the status information from the device, thus reducing the signaling overhead.

The main contributions of this paper are summarized as follows:

1. We propose an approach to offloading DNN-based sensing tasks that adapts to dynamic on-device and edge server workloads, enabling offloading decision-making to optimize task performance.
2. We propose a learning-assisted algorithm, which generates effective offloading decisions with unknown task arrival statistics. Empowered by DTs, the algorithm can promptly attain an effective offloading solution with sufficient training data.

3. We derive necessary conditions for optimal offloading decisions and accordingly use them for decision space reduction, which reduces the complexity for adaptive offloading of DNN-based sensing tasks.

The remainder of the chapter is organized as follows. In Chapter 4.2, we introduce the system model. In Chapter 4.3, a DT-assisted approach to adaptive device-edge collaboration on DNN inference is proposed. The problem is formulated and transformed in Chapter 4.4, and the DT and learning-assisted offloading decision-making algorithm is proposed in Chapter 4.5. In Chapter 4.6, we investigate offloading decision space reduction. Simulation results are provided in Chapter 4.7. Chapter 4.8 concludes the chapter.

4.2 System Model

In this section, we first introduce the models of DNN task generation, computing and queuing. Then, we define the utility of a DNN task, which incorporates the delay, inference accuracy, and energy consumption for processing a DNN task.

4.2.1 Task Generation Model

Consider an AIoT device that connects to an edge computing server through an AP. The AIoT device collects sensing data of individual targets using the mono-static sensing mode and generates computing tasks such as object recognition for processing the collected sensing data. The time horizon is divided into slots with equal duration denoted by ΔT , and the index of a time slot is denoted by $t \in \{1, 2, \dots\}$. At the beginning of each time slot, a task is probabilistically generated by the device with an unknown probability [91]. Let ΔT_n represent the interval between the time instants when the $(n + 1)$ -th and the n -th tasks are generated, where $n \in \{1, 2, \dots, N\}$, and let ΔT_0 represent the time instant when the first task is generated.

4.2.2 Task Computing Model

For processing the tasks, one full-size DNN with L consecutive layers (e.g., $L = 7$ in Fig. 4.1) is deployed at the edge server, and one shallow DNN with fewer layers is deployed at the AIoT device. For DNNs that contain parallel layers or residual blocks, the layers in each residual block or the layers with parallel execution can be abstracted as one logical

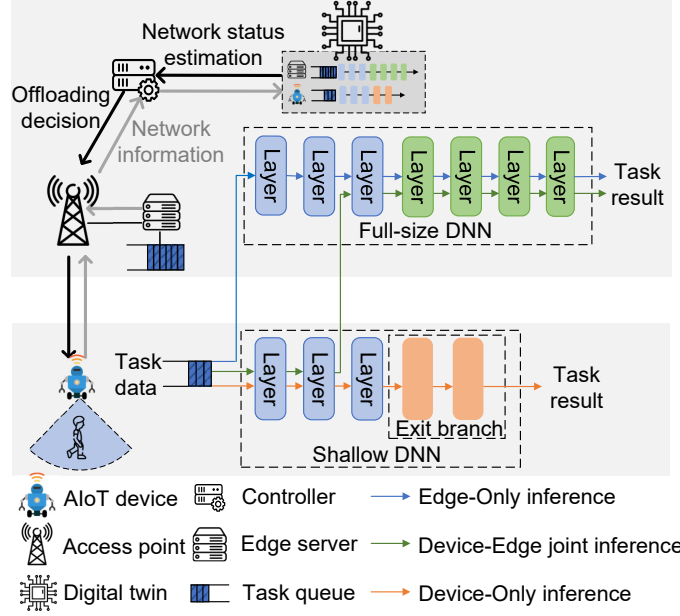


Fig. 4.1: DNN-based sensing data processing model.

layer [137, 89]. In the design and training of the full-size and shallow DNNs, the BranchyNet architecture [138] is used such that the first l_e layers of the two DNNs are identical (e.g., the three blue-colored layers in Fig. 4.1). The remaining part of the shallow DNN is referred to as an exit branch, and the layers therein (e.g., the two orange-colored layers in Fig. 4.1) are abstracted as one logical layer, i.e., the $(l_e + 1)$ -th layer of the shallow DNN.

Each time before the AIoT device executes the next layer in the shallow DNN, a network controller decides whether the AIoT device should stop locally processing the task at this point and offload the task to the edge server instead. For the n -th task, when the task is offloaded to the edge server or locally completed, the number of executed layers in the shallow DNN for the task is determined, which is denoted by $x_n \in \{0, 1, \dots, l_e + 1\}$. We define x_n as the offloading decision for the n -th task, which can result in three DNN inference scenarios as illustrated in Fig. 4.1:

1. *Edge-Only Inference*: If $x_n = 0$, the n -th DNN task is offloaded to the edge server without being processed by any layer of the shallow DNN by the device. In other words, the task is completely processed by the full-size DNN at the edge server.
2. *Device-Edge Joint Inference*: If $1 \leq x_n \leq l_e$, the n -th DNN task is locally processed by the first x_n layers of the shallow DNN (e.g., $x_n = 2$ for the device-edge joint

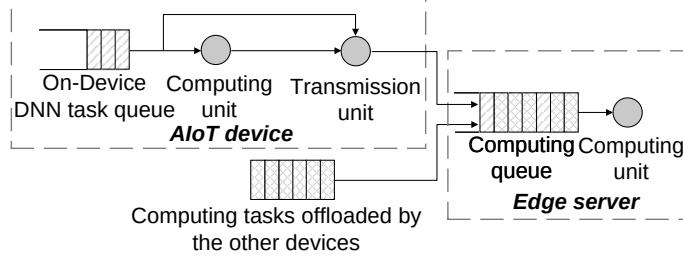


Fig. 4.2: Task queuing model.

inference scenario in Fig. 4.1). Then, the output of the x_n -th layer of the shallow DNN, as the intermediate result, is uploaded to the edge server and processed by the remaining layers of the full-size DNN.

3. *Device-Only Inference*: If $x_n = l_e + 1$, the n -th DNN task is not offloaded to the edge server but completely processed by the shallow DNN at the device.

To assist in making offloading decisions of DNN tasks, as shown in Fig. 4.1, DTs are constructed by the network controller. Specifically, the network controller collects the network information, i.e., the per-layer on-device DNN inference delay and the information on task arrivals at the AIoT device and the edge server. The collected information is processed by DT models to estimate the status of the AIoT device and the edge server for assisting in making offloading decisions of DNN tasks. Detailed introduction of the DTs will be presented in Section 4.3.

4.2.3 Task Queuing Model

As shown in Fig. 4.2, the task queuing model includes queuing at the device and queuing at the edge server.

1) *On-Device Task Queue*: The device has one computing unit and one transmission unit, which can respectively process and transmit only one task at any time instant. The tasks that have been generated but not yet processed or offloaded are stored in an on-device task queue following the First-Come-First-Serve (FCFS) rule. When the computing unit becomes idle, the first task in the on-device task queue leaves the queue. If the transmission unit is also idle, edge-only inference, i.e., the transmission unit offloads the task to the edge server without any on-device processing, can be chosen for the task. Otherwise, the task will be processed by the computing unit until the transmission unit is idle and a decision

to offload the task is made. At the beginning of the $(t+1)$ -th time slot, the on-device computing workload in terms of the number of tasks in the on-device queue, denoted by $Q^D(t+1)$, is updated as:

$$Q^D(t+1) = Q^D(t) + I(t+1) - O(t+1), \quad (4.1)$$

where $I(t)$ equals to 1 if one DNN task is generated in the $(t-1)$ -th time slot and 0 otherwise; $O(t)$ equals to 1 if a DNN task leaves the on-device task queue at the beginning of the t -th time slot and 0 otherwise.

2) *Edge Server Queue*: The edge server has one computing unit, which can process only one task at any time instant, and maintains a computing queue for tasks yet to be processed. The computing tasks arriving at the edge server in each time slot will either start to be processed by the computing unit or enter the computing queue at the beginning of the next time slot.¹ The edge server workload is the sum of CPU cycles required to complete the processing of the task in the computing unit as well as all the tasks in the computing queue. At the beginning of the $(t+1)$ -th time slot, the edge server workload, denoted by $Q^E(t+1)$, is updated as:

$$Q^E(t+1) = \max\{Q^E(t) - f^E \Delta T, 0\} + D(t) + W(t), \quad (4.2)$$

where $D(t)$ and $W(t)$ represent the workload of the tasks from the considered device and other devices connected to the edge server, respectively, in the t -th time slot; f^E represents the computation frequency of the edge server in the unit of CPU cycles per second.

4.2.4 Task Utility

In this subsection, we first derive the delay, inference accuracy, and energy consumption of processing a DNN task. Then, we define the utility of a DNN task.

1) *Delay*: Depending on the task offloading decision, the processing of a DNN task can incur *i*) on-device queuing delay, *ii*) on-device inference delay, *iii*) transmission delay from uploading the intermediate result, *iv*) queuing delay at the edge server, and *v*) edge inference delay. The delay from delivering the task result to the device is neglected due to the typically small size of the result.

¹The processing order of the computing tasks arriving at the edge server in a time slot is determined by a task scheduling algorithm. For simplicity, we assume that the task offloaded by the considered device will be first processed.

i) *On-Device Inference Delay*: The on-device inference delay of the n -th task is the sum of the execution delay of the first x_n layers in the shallow DNN, given by:

$$T_n^{\text{lc}}(x_n) = \begin{cases} \sum_{l=1}^{x_n} d_l^D, & x_n \geq 1, \\ 0, & x_n = 0, \end{cases} \quad (4.3)$$

where d_l^D represents the time to execute the l -th layer in the shallow DNN by the device, $\forall l \in \{1, 2, \dots, l_e + 1\}$. The value of d_l^D is rounded to an integer multiple of the time slot duration ΔT .

ii) *On-Device Queuing Delay*: The queuing delay of the n -th DNN task at the device is determined by the offloading decisions for the DNN tasks generated earlier, which are denoted as $\mathbf{x}_{n-1} = \{x_1, x_2, \dots, x_{n-1}\}$. Let $\mathbf{x}_0 = \emptyset$ for consistency. The on-device queuing delay for the n -th task, denoted by $T_n^{\text{lc}}(\mathbf{x}_{n-1})$, can be calculated by:

$$T_n^{\text{lc}}(\mathbf{x}_{n-1}) = \begin{cases} 0, & n = 1, \\ \max\{T_{n-1}^{\text{lc}}(\mathbf{x}_{n-2}) + T_{n-1}^{\text{lc}}(x_{n-1}) - \Delta T_{n-1}, 0\}, & n \geq 2. \end{cases} \quad (4.4)$$

iii) *Data Uploading Delay*: If edge-only inference or device-edge joint inference is adopted for the n -th DNN task, i.e., $0 \leq x_n \leq l_e$, the task processing will incur an uploading delay denoted by T_n^{up} . The data to be uploaded is the input to the $(x_n + 1)$ -th layer of the shallow DNN, and the uploading delay can be calculated by:

$$T_n^{\text{up}}(x_n) = \begin{cases} s_{x_n}/R_0, & 0 \leq x_n \leq l_e, \\ 0, & x_n = l_e + 1, \end{cases} \quad (4.5)$$

where s_l represents the size of the data to the $(l + 1)$ -th layer of the shallow DNN, $\forall l \in \{0, 1, \dots, l_e\}$; R_0 is the uplink transmission rate between the device and the AP.

iv) *Edge Queuing Delay*: If the n -th DNN task is offloaded to the edge server, i.e., $x_n \leq l_e$, the task processing can incur a queuing delay at the edge server, given by:

$$T_n^{\text{eq}}(\mathbf{x}_n) = \begin{cases} Q^E(t_{n,x_n})/f_s^E, & 0 \leq x_n \leq l_e, \\ 0, & x_n = l_e + 1, \end{cases} \quad (4.6)$$

where t_{n,x_n} is the index of the time slot when the n -th DNN task arrives at the edge server.²

² Assuming a high data transmission rate, the data uploading for a DNN task can be completed within one time slot.

v) *Edge Inference Delay*: If $x_n \leq l_e$, the task processing will incur an inference delay at the edge server from executing the remaining layers of the full-size DNN, given by:

$$T_n^{\text{ec}}(x_n) = \begin{cases} \sum_{l=x_n+1}^L d_l^E, & 0 \leq x_n \leq l_e, \\ 0, & x_n = l_e + 1, \end{cases} \quad (4.7)$$

where d_l^E represents the execution time in the l -th layer of the full-size DNN by the edge server.

Summarizing the delay in equations (4.3) to (4.7), the overall delay of the n -th DNN task, denoted by $T_n(\mathbf{x}_n)$, can be calculated by:

$$T_n(\mathbf{x}_n) = T_n^{\text{lc}}(\mathbf{x}_{n-1}) + T_n^{\text{lc}}(x_n) + T_n^{\text{up}}(x_n) + T_n^{\text{eq}}(\mathbf{x}_n) + T_n^{\text{ec}}(x_n). \quad (4.8)$$

2) *Inference Accuracy*: The accuracy of the task result is denoted by $A_n(x_n)$, which equals η^E if the task is offloaded to the edge server for processing by the full-size DNN, i.e., $1 \leq x_n \leq l_e$, and η^D if the result is derived by the shallow DNN, i.e., $x_n = l_e + 1$. The accuracy of the task result derived by the full-size DNN is higher than that derived by the shallow DNN, i.e., $\eta^E > \eta^D$.

3) *Energy Consumption*: The energy consumption for processing the n -th DNN task, denoted by $E_n(x_n)$, includes two parts, i.e., the energy consumption for DNN inference and that for task data uploading. The overall energy consumption can be calculated by:

$$E_n(x_n) = \kappa^D (f_s^D)^3 T_n^{\text{lc}} + \kappa^E (f_s^E)^3 T_n^{\text{ec}} + p^{\text{up}} T_n^{\text{up}}, \quad (4.9)$$

where κ^D and κ^E are the energy efficiency coefficients of the device and the edge server, respectively [82]; p^{up} is the transmit power of the device.

4) *Task Utility*: We define the utility of the n -th DNN task, denoted by $U_n(\mathbf{x}_n)$, as a weighted sum of the overall delay, inference accuracy, and energy consumption of the task:

$$U_n(\mathbf{x}_n) = -T_n(\mathbf{x}_n) + \alpha A_n(x_n) - \beta E_n(x_n), \quad (4.10)$$

where α and β are positive weights for the inference accuracy and energy consumption, respectively.

4.3 DT-Assisted Adaptive Device-Edge Collaboration on DNN Inference

In this section, we first introduce the data required to establish the DTs of two processes, i.e., a DT of the on-device DNN inference and a DT of the computing workload evolution. Then, we introduce how the DT of on-device inference can be established to estimate the on-device inference status, and how the DT of computing workload evolution can be established to emulate the on-device workload and edge server workload, respectively. Finally, assisted by the two DTs, an approach to adaptive device-edge collaboration on DNN inference is proposed.

4.3.1 Data for Establishing DTs

The following information for establishing the DTs is estimated or collected by the network controller:

- *Per-Layer On-Device Inference Delay:* The delay for executing each layer of the shallow DNN by the device, i.e., $\{d_l^D, l = 1, 2, \dots, l_e + 1\}$, is estimated by the network controller. Specifically, the estimation can be based on *i)* the number of FLOPs required for each layer and the computing capability of the device [139]; or *ii)* regression models given the configuration of each layer such as the input and output data sizes [34].
- *Task Generation and Arrival:* At the beginning of each time slot, the controller collects information on *i)* whether a DNN task is generated by the AIoT device at the beginning of each time slot, i.e., $\{I(t), t = 1, 2, \dots\}$; *ii)* the workload of computing tasks arriving at the edge server not from the considered device in the time slot, i.e., $\{W(t), t = 1, 2, \dots\}$.

4.3.2 DT of On-Device DNN Inference

For each DNN task, the network controller determines whether to continue the on-device inference before the device executes each layer in the shallow DNN. As a result, the change of on-device DNN inference status, i.e., a layer is about to be locally executed, needs to be known by the network controller. Instead of acquiring the on-device inference status from the device in real-time, which can result in large signaling overhead, the network

controller establishes a DT to emulate the on-device inference and estimate the on-device inference status. Denote the index of the time slot right before the on-device execution of the $(l + 1)$ -th layer for the n -th DNN task by $t_{n,l}, \forall l \in \{0, 1, \dots, l_e\}$. Then, $t_{n,l}$ can be estimated based on the time instant when the n -th DNN task is generated, the on-device queuing delay of the n -th DNN task, and the on-device inference delay before executing the $(l + 1)$ -th layer:

$$t_{n,l}(\mathbf{x}_{n-1}) = \frac{1}{\Delta T} \left(\left(\sum_{i=0}^{n-1} \Delta T_i \right) + T_n^{\text{Iq}}(\mathbf{x}_{n-1}) + \sum_{i=0}^l d_i^D \right). \quad (4.11)$$

In addition, $t_{n,l_e+1}(\mathbf{x}_{n-1})$ is also calculated, which represents the index of the next time slot if the task were completed by device-only inference.

4.3.3 DT of Computing Workload Evolution

The evolution of the computing workload on the AIoT device and the edge server affects the average utility of the DNN tasks. To capture the evolution with unknown statistics, a learning algorithm can be applied when determining whether to continue the on-device inference for a DNN task. Such an algorithm can be trained based on the utility of the tasks and the decisions made for the tasks. However, if a decision to offload a task is made, options of continuing the on-device inference for the task cannot be evaluated for training the algorithm. This is because offloading the task would affect the workload evolution, which makes the workload evolution if the on-device inference continued unobservable.

To estimate the on-device and edge server computing workload status under each candidate offloading decision for a DNN task, the DT of computing workload evolution is established. Specifically, for each task, the DT emulates the workload evolution in a hypothetical case, i.e., if the task processing were completed by the shallow DNN at the device. The time slots to execute the shallow DNN for the n -th task are denoted by $t \in \{t_{n,0}, t_{n,0} + 1, \dots, t_{n,l_e+1} - 1\}$, and the emulated on-device workload and edge server workload, denoted by $\tilde{Q}^D(t)$ and $\tilde{Q}^E(t)$, respectively, are calculated by:

$$\tilde{Q}^D(t) = \begin{cases} Q^D(t), & t = t_{n,0}, \\ \tilde{Q}^D(t-1) + I(t), & \text{otherwise,} \end{cases} \quad (4.12a)$$

$$\tilde{Q}^E(t) = \begin{cases} Q^E(t), & t = t_{n,0}, \\ \max\{\tilde{Q}^E(t-1) - f^E \Delta T, 0\} + W(t), & \text{otherwise.} \end{cases} \quad (4.12b)$$

The difference between the actual computing workloads in (4.1) and (4.2) and the emulated computing workloads in (4.12) is that the former are calculated based on the actual offloading decision of a DNN task while the latter are calculated in the hypothetical case that the task processing were locally completed. In such a hypothetical case, the decrease of the computation workload at the device due to the offloading of the task, i.e., $O(t)$ in (4.1), and the increase of the computation workload at the edge server due to the offloading of the task, i.e., $D(t)$ in (4.2), both equal zero. Correspondingly, equation (4.12) for calculating the emulated workloads in the hypothetical case omits $O(t)$ and $D(t)$, which is the major difference with (4.1) and (4.2).

4.3.4 Analysis of DTs' Overheads

The data collection and processing for DTs can result in a delay and resource consumption. Nevertheless, such a delay and resource consumption are relatively low. This is because, first, the data collected for DTs as described in subsection 4.3.1 have a small size. Second, the processing of the collected data, following (4.11) and (4.12), only involves several addition operations. Third, only the recently collected data, i.e., the information on tasks that arrive at the AIoT device and the edge server during the on-device processing of the current DNN task, are required for data processing. As a result, the outdated data, such as the information on tasks that arrive during the on-device processing of the preceding DNN tasks, can be discarded to minimize the cost of storing the DTs.

4.3.5 DT-Assisted Approach to Adaptive Device-Edge Collaboration

As shown in Fig. 4.3, with the DT of on-device inference and the DT of computing workload evolution, we propose an approach to adaptive device-edge collaboration on DNN inference, which involves the following four steps.

- *Step 1 (Task Information Gathering)*: In the beginning of each time slot, the device sends the DNN task generation indicator $I(t)$ to the controller, which then calculates the gaps of task generations $\{\Delta T_n, n = 0, 1, \dots, (\sum_t I(t)) - 1\}$. After the controller determines the offloading decision for the $(n-1)$ -th DNN task, i.e., x_{n-1} , the on-device queuing delay of the n -th DNN task, i.e., $T_n^{\text{dq}}(\mathbf{x}_{n-1})$, is calculated. Then, the controller utilizes the DT of on-device inference and (4.11) to estimate the indices of the time slots when the device would execute each layer of the shallow DNN for the n -th task, i.e., $\{t_{n,l}(\mathbf{x}_n)\}_{l=0}^{l_e}$, assuming the task continues to be processed locally.

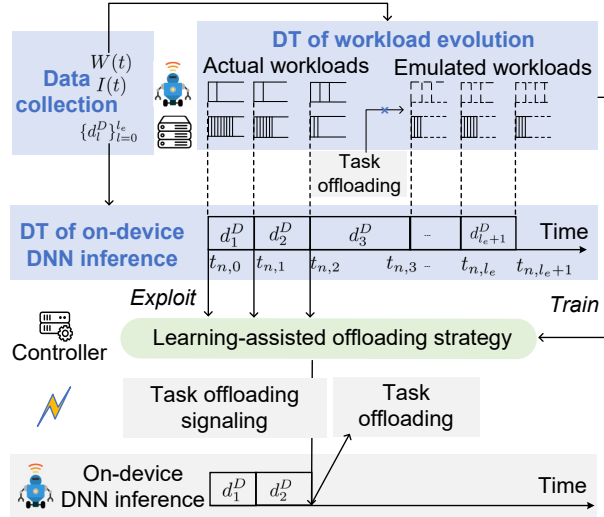


Fig. 4.3: DT-Assisted approach to adaptive device-edge collaboration on DNN inference.

- *Step 2 (Learning-Assisted Offloading Decision-Making):* After the transmission unit is idle, i.e., $t \geq t_{n,\hat{x}_n}$, at the beginning of any time slot $t \in \{t_{n,l}(\mathbf{x}_{n-1})\}_{l=\hat{x}_n}^{l_e}$, leveraging a learning-assisted algorithm, the controller determines whether the device should continue the on-device execution of the $(l+1)$ -th layer of the shallow DNN or upload the intermediate result to the edge server.
- *Step 3 (Signaling of Task Offloading):* If the controller determines to stop the on-device inference for the n -th DNN task in the beginning of the t -th time slot, it sends a signal to the device, corresponding to setting $O(t) = 1$, to the device. The device then uploads the intermediate result to the edge server for executing the remaining layers of the full-size DNN. Without receiving this signal, the device will continue executing the next layer of the shallow DNN.
- *Step 4 (Training of Learning-Assisted Offloading Algorithm):* After the time slot $t = t_{n,l_e+1}(\mathbf{x}_{n-1})$, the workload evolution if the n -th DNN task were locally completed by the shallow DNN, is generated using the DT of computing workload evolution and (4.12). Then, a learning-assisted offloading algorithm will be trained with the data augmented by the emulated workload evolution.

4.4 Problem Formulation and Transformation

In this section, we formulate the problem of maximizing the average utility of DNN tasks generated by the considered AIoT device. Due to the stochastic DNN task generation, we transform the problem into multiple sub-problems, which are solved sequentially to make the offloading decision for each task given the decisions for its preceding tasks.

4.4.1 Problem Formulation

We aim to maximize the average utility of DNN tasks generated at the considered device by optimizing the offloading decisions for the tasks. The problem is given by:

$$(P1) \max_{\mathbf{x}_N} \lim_{N \rightarrow \infty} \frac{1}{N} \sum_{n=1}^N U_n(\mathbf{x}_n) \quad (4.13a)$$

$$\text{s.t. } 0 \leq x_n \leq l_e + 1, \quad (4.13b)$$

$$\begin{aligned} & (\sum_{i=m}^{n-1} \Delta T_i) + T_n^{\text{lc}}(\mathbf{x}_{n-1}) + T_n^{\text{lc}}(x_n) \geq T_m^{\text{lc}}(\mathbf{x}_{m-1}) \\ & + T_m^{\text{lc}}(x_m) + T_m^{\text{up}}(x_m), \forall m \in \{1, \dots, n-1\}, n \in \{1, 2, \dots, N\}, \end{aligned} \quad (4.13c)$$

where (4.13c) ensures that the n -th DNN task can be offloaded to the edge server only when the offloading of preceding DNN tasks, if any, has been completed and the transmission unit is idle. Note that $T_n^{\text{lc}}(x_n)$ monotonically increases with x_n . As a result, given any $\mathbf{x}_{n-1}$, we can find for the n -th task a minimum number of layers in the shallow DNN, denoted by $\hat{x}_n(\mathbf{x}_{n-1})$, which should be locally executed to satisfy (4.13c). An equivalent form of constraints (4.13b) and (4.13c) is:

$$\hat{x}_n(\mathbf{x}_{n-1}) \leq x_n \leq l_e + 1, \forall n \in \{1, 2, \dots, N\}. \quad (4.14)$$

4.4.2 Problem Transformation

Solving (P1) requires the joint optimization of offloading decisions for all the DNN tasks, which is challenging due to the stochastic task generation at the device with unknown statistics [91]. In this subsection, we first define the long-term utility of one task, which incorporates the main impact of the offloading decision of the task on the other tasks, i.e., the on-device queuing delay of the other tasks resulting from the on-device inference of

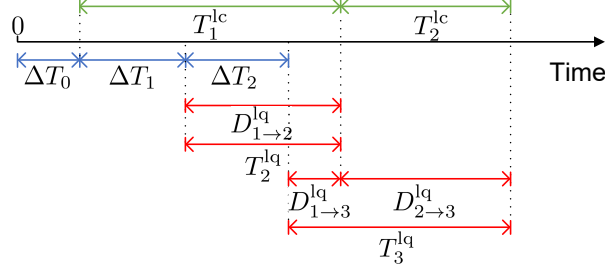


Fig. 4.4: Decomposition of the on-device queuing delay T_n^{lq} .

the task.³ Then, we transform (P1) into the online optimization of the offloading decision for each DNN task given the offloading decisions of the preceding DNN tasks, with the objective to maximize the expected long-term utility of each DNN task.

1) *Long-Term Utility of DNN Task*: The on-device queuing delay of a DNN task results from the on-device inference for its preceding tasks. As shown by the red lines in Fig. 4.4, we first decompose the on-device queuing delay of a task as the summation of the queuing delay resulting from each of its preceding tasks. Then, we define the long-term on-device queuing delay for processing each task, which is the sum on-device queuing delay of subsequent tasks resulting from the on-device processing of the task. For brevity, we omit the parentheses and the offloading decision variables in notations.

First, define $D_{m \rightarrow n}^{\text{lq}}$ as the on-device queuing delay of the n -th DNN task directly resulting from the on-device inference of the m -th DNN task, calculated by:

$$D_{m \rightarrow n}^{\text{lq}} = \begin{cases} 0, & m \geq n, \\ \max\{\min\{T_m^{\text{lq}} - \sum_{i=m}^{n-1} \Delta T_i, 0\} + T_m^{\text{lc}}, 0\}, & m < n. \end{cases} \quad (4.15)$$

When $m \geq n$, $D_{m \rightarrow n}^{\text{lq}}$ equals to zero since tasks leave the on-device task queue following the FCFS rule; when $m < n$, the maximum $D_{m \rightarrow n}^{\text{lq}}$ is the on-device inference delay of the m -th task, i.e., T_m^{lc} (e.g., $D_{2 \rightarrow 3}^{\text{lq}} = T_2^{\text{lc}}$ in Fig. 4.4). Given the offloading decision of the m -th DNN task, $D_{m \rightarrow n}^{\text{lq}}$ decreases with the increase of the gap between the generation instants of the m -th and the n -th tasks (e.g., $D_{1 \rightarrow 3}^{\text{lq}}$ decreases with the increase of ΔT_1 or ΔT_2 in Fig. 4.4).

Proposition 4. *The on-device queuing delay of the n -th DNN task is the sum of the*

³Although the edge queuing delay of a DNN task from the considered device can be affected by the offloading decisions of its preceding DNN tasks, such effect is relatively small as the edge queuing delay is affected by the computing tasks offloaded to the edge server from the other devices.

on-device queuing delay resulting from all the N DNN tasks generated by the device:

$$T_n^{\text{lc}} = \sum_{m=1}^N D_{m \rightarrow n}^{\text{lc}}. \quad (4.16)$$

Proof. See Appendix D. $\square$

Proposition 5. Define the queuing delay of all N DNN tasks due to the on-device inference for the n -th DNN task, i.e., $D_n^{\text{lc}} = \sum_{m=1}^N D_{n \rightarrow m}^{\text{lc}}$, as the long-term on-device queuing delay of the n -th DNN task, which can be calculated as follows:

$$D_n^{\text{lc}} = \begin{cases} 0, & T_n^{\text{lc}} = 0, \\ \sum_{t=t_{n,0}}^{t_{n,0}+(T_n^{\text{lc}}/\Delta T)-1} Q^D(t)\Delta T, & \text{otherwise.} \end{cases} \quad (4.17)$$

Proof. See Appendix E. $\square$

Equation (4.17) can be interpreted as follows. First, if a DNN task is directly offloaded to the edge server without any on-device inference, i.e., $T_n^{\text{lc}} = 0$, the on-device queuing delay of the other tasks would not be increased, i.e., $D_n^{\text{lc}} = 0$. Otherwise, if the device is processing the task in the t -th time slot, the on-device queuing delay of each task in the on-device task queue will be increased by the time duration of a time slot, i.e., ΔT . As a result, the sum of the increased on-device queuing delay for the tasks in the on-device queue in the t -th time slot is $Q^D(t)\Delta T$. The increased queuing delay of the subsequent tasks due to the on-device inference for the n -th task is thus the summation term in (4.17).

By substituting $T_n^{\text{lc}}(x_n)$ in the overall delay (4.8) with $D_n^{\text{lc}}(x_n)$, we define the time cost of the n -th task as:

$$C_n(\mathbf{x}_n) = D_n^{\text{lc}}(\mathbf{x}_n) + T_n^{\text{lc}}(x_n) + T_n^{\text{up}}(x_n) + T_n^{\text{eq}}(\mathbf{x}_n) + T_n^{\text{ec}}(x_n), \quad (4.18)$$

where D_n^{lc} is a function of $\mathbf{x}_n$ since $t_{n,0}$ and T_n^{lc} in (4.17) are functions of $\mathbf{x}_{n-1}$ and x_n , respectively.

By substituting $T_n(\mathbf{x}_n)$ in (4.10) with $C_n(\mathbf{x}_n)$, we define the long-term utility of the n -th DNN task by:

$$U_n^{\text{lt}}(\mathbf{x}_n) = -C_n(\mathbf{x}_n) + \alpha A_n(x_n) - \beta E_n(x_n). \quad (4.19)$$

Compared with the original utility of a DNN task in (4.10), which incorporates the on-device queuing delay of the task, the long-term utility in (4.19) incorporates the on-device queuing delay of subsequent tasks due to the on-device inference for the task. As a

result, when sequentially and individually maximizing the long-term utility of each task, the on-device queuing delay of subsequent tasks can be minimized accordingly.

Based on *Proposition 4*, the sum of the on-device queuing delay of all the N DNN tasks can be calculated by:

$$\begin{aligned}\sum_{n=1}^N D_n^{\text{dq}} &= \sum_{n=1}^N (\sum_{m=1}^N D_{n \rightarrow m}^{\text{dq}}) \\ &= \sum_{n=1}^N (\sum_{m=1}^N D_{m \rightarrow n}^{\text{dq}}) \\ &= \sum_{n=1}^N T_n^{\text{dq}}.\end{aligned}\tag{4.20}$$

Thus, we have:

$$\sum_{n=1}^N U_n(\mathbf{x}_n) = \sum_{n=1}^N U_n^{\text{lt}}(\mathbf{x}_n),\tag{4.21}$$

which indicates that the average utility of the N DNN tasks, as the objective in (P1), equals to the average long-term utility of the N DNN tasks.

2) *Problem Transformation*: Due to the stochastic DNN task generation at the considered device and the task arrival at the edge server, we transform (P1) into the sequential and individual maximization of the *expected* long-term utility of a task given the offloading decisions of all preceding tasks:

$$(P2) \quad \max_{\hat{x}_n(\mathbf{x}_{n-1}) \leq x_n \leq l_e + 1} \mathbb{E} [U_n^{\text{lt}}(\mathbf{x}_n)].\tag{4.22}$$

4.5 DT and Learning-Assisted Algorithm for Offloading decision-making

In this section, to solve the transformed problem (P2), we propose an algorithm for DNN task offloading decision-making, assisted by DTs and machine learning techniques as shown in Fig. 4.5. In this algorithm, we use the optimal stopping theorem and a neural network to make offloading decisions, and propose DT-assisted training for the neural network.

For brevity, we denote $D_n^{\text{dq}}(x_n = l | \mathbf{x}_{n-1})$, $T_n^{\text{eq}}(x_n = l | \mathbf{x}_{n-1})$ and $U_n^{\text{lt}}(x_n = l | \mathbf{x}_{n-1})$ by D_l^{dq} , T_l^{eq} and U_l^{lt} , which respectively represent the long-term on-device queuing delay, the edge queuing delay and the long-term task utility if the offloading decision of the n -th DNN task is l , i.e., $x_n = l$, given that the offloading decisions of the first $n-1$ DNN tasks are $\mathbf{x}_{n-1}$. To emphasize the impact of the long-term on-device queuing delay and the edge queuing delay on the long-term utility of a DNN task, we represent U_l^{lt} as $U_l^{\text{lt}}(D_l^{\text{dq}}, T_l^{\text{eq}})$. In addition, we define $\mathbf{D}_l^{\text{dq}} = \{D_0^{\text{dq}}, D_1^{\text{dq}}, \dots, D_l^{\text{dq}}\}$ and $\mathbf{T}_l^{\text{eq}} = \{T_0^{\text{eq}}, T_1^{\text{eq}}, \dots, T_l^{\text{eq}}\}$.

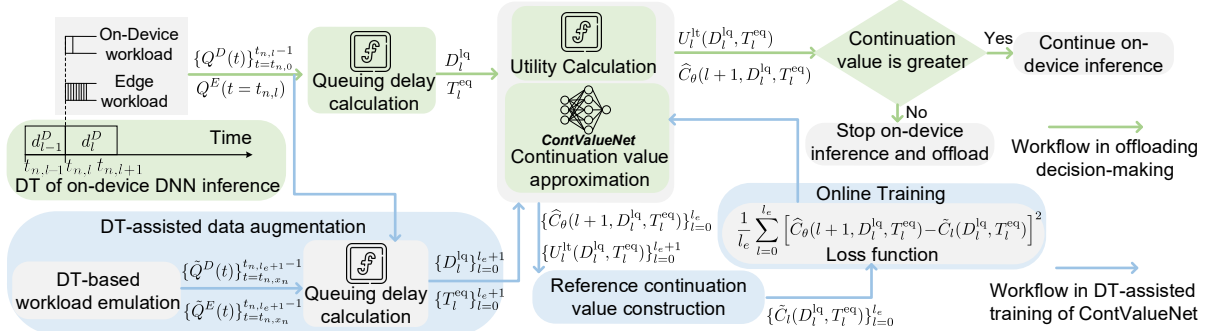


Fig. 4.5: DT and learning-assisted algorithm for DNN task offloading decision-making.

4.5.1 DT and Learning-Assisted Offloading Decision-Making

1) *Continuation Value-based Offloading Decision-Making*: Define $V_l(\mathbf{D}_l^{lq}, \mathbf{T}_l^{eq})$ as the *maximum* expected long-term utility of a DNN task when l layers of the shallow DNN have been executed for the task, where $l \in \{0, 1, \dots, l_e + 1\}$. Such a value satisfies a recursion rule:

$$V_l(\mathbf{D}_l^{lq}, \mathbf{T}_l^{eq}) = \begin{cases} \max \{U_l^{lt}(D_l^{lq}, T_l^{eq}), C_l(\mathbf{D}_l^{lq}, \mathbf{T}_l^{eq})\}, & l \leq l_e, \\ U_{l_e+1}^{lt}(D_{l_e+1}^{lq}, T_{l_e+1}^{eq}), & l = l_e + 1, \end{cases} \quad (4.23)$$

where $\forall l \in \{0, 1, \dots, l_e\}$, we have

$$C_l(\mathbf{D}_l^{lq}, \mathbf{T}_l^{eq}) = \mathbb{E} [V_{l+1}(\mathbf{D}_{l+1}^{lq}, \mathbf{T}_{l+1}^{eq}) | \mathbf{D}_l^{lq}, \mathbf{T}_l^{eq}], \quad (4.24)$$

referred to as the *continuation value*[140].

Proposition 6. *Based on optimal stopping theorem [141], the optimal offloading decision for maximizing the expected long-term utility of the n -th DNN task is:*

$$x_n = \begin{cases} \min\{\Psi\}, & \text{if } \Psi \neq \emptyset, \\ l_e + 1, & \text{otherwise,} \end{cases} \quad (4.25a)$$

where

$$\Psi = \left\{ l \mid \hat{x}_n \leq l \leq l_e, U_l^{lt}(D_l^{lq}, T_l^{eq}) \geq C_l(\mathbf{D}_l^{lq}, \mathbf{T}_l^{eq}) \right\}. \quad (4.25b)$$

The continuation value-based offloading decision-making given in (4.25) is explained as follows. For the n -th DNN task, after the device executes l layers of the shallow DNN,

where $l \in \{\hat{x}_n, \hat{x}_n + 1, \dots, l_e\}$, if the long-term utility of offloading the DNN task now, i.e., $U_l^{\text{lt}}(D_l^{\text{q}}, T_l^{\text{eq}})$, is no less than the continuation value, i.e., $C_l(\mathbf{D}_l^{\text{q}}, \mathbf{T}_l^{\text{eq}})$, the on-device inference for the task should be stopped for offloading the task to the edge server. Otherwise, the device should continue executing the $(l + 1)$ -th layer of the shallow DNN.

2) *DT and Learning-Assisted Offloading Decision-Making*: The continuation value for each layer of the shallow DNN can be obtained using backward induction [141], which however requires the prior statistics of the workload evolution and introduces computing overhead [142]. Inspired by [140], our approach is to construct a neural network referred to as ContValueNet to generate an approximated continuation value $\hat{C}_\theta(l + 1, D_l^{\text{q}}, T_l^{\text{eq}})$ given the input $\{l + 1, D_l^{\text{q}}, T_l^{\text{eq}}\}$, where θ represents the parameters in ContValueNet. By substituting the continuation value in (4.25b) with the approximated continuation value, a learning-assisted offloading algorithm is obtained. Specifically, as shown in the green blocks in Fig. 4.5, through the DT of on-device DNN inference given in (4.11), the controller is informed when a layer of the shallow DNN is about to be executed at the device. Before the on-device execution of the $(l + 1)$ -th layer by the device, the controller calculates *i*) the queuing delay D_l^{q} and T_l^{eq} ; *ii*) calculates the long-term utility of the task $U_l^{\text{lt}}(D_l^{\text{q}}, T_l^{\text{eq}})$; and *iii*) approximates the continuation value by the ContValueNet, i.e., $\hat{C}_\theta(l + 1, D_l^{\text{q}}, T_l^{\text{eq}})$. If the continuation value is greater than the utility, the controller lets the device execute the next layer for the task. Otherwise, the network controller lets the device upload the input data of the $(l + 1)$ -th layer in the full-size DNN to the edge server.

4.5.2 DT-Assisted Training of ContValueNet

As shown in the blue blocks in Fig. 4.5, DTs are leveraged to assist in training ContValueNet, which involves three steps, i.e., DT-assisted data augmentation, reference continuation value construction, and online ContValueNet training.

1) *DT-Assisted Data Augmentation*: If the n -th DNN task is determined to offload to the edge server, i.e., $x_n \leq l_e$, the controller emulates the long-term on-device queuing delay and the edge queuing delay given the remaining potential offloading decisions for the task, i.e., $\{D_l^{\text{q}}\}_{l=x_n+1}^{l_e+1}$ and $\{T_l^{\text{eq}}\}_{l=x_n+1}^{l_e+1}$, using the DT of workload evolution and (4.12). Specifically, $\{D_l^{\text{q}}\}_{l=x_n+1}^{l_e}$ are calculated by substituting $Q^D(t)$ in (4.17) with $\tilde{Q}^D(t)$ in (4.12a), and $\{T_l^{\text{eq}}\}_{l=x_n+1}^{l_e+1}$ are calculated by substituting the on-device workload in (4.6) with the emulated on-device workload in (4.12b). Then, the controller calculates the approximated continuation values $\{\hat{C}_\theta(l + 1, D_l^{\text{q}}, T_l^{\text{eq}})\}_{l=x_n+1}^{l_e+1}$ using ContValueNet. Combining the above values with $\{D_l^{\text{q}}\}_{l=0}^{x_n}$, $\{T_l^{\text{eq}}\}_{l=0}^{x_n}$ and $\{\hat{C}_\theta(l + 1, D_l^{\text{q}}, T_l^{\text{eq}})\}_{l=0}^{x_n}$ which are collected during the

decision-making for the n -th DNN task, the controller obtains $\{D_l^{\text{dq}}\}_{l=0}^{l_e+1}$, $\{T_l^{\text{eq}}\}_{l=0}^{l_e+1}$ and $\{\widehat{C}_\theta(l+1, D_l^{\text{dq}}, T_l^{\text{eq}})\}_{l=0}^{l_e+1}$, which will be used to construct the reference continuation values.

2) *Reference Continuation Value Construction*: To optimize the parameters θ of ContValueNet, reference continuation values of evaluating the approximation error of ContValueNet should be constructed [140].

First, for any $l \in \{0, 1, 2, \dots, l_e\}$, we calculate $\widehat{V}_l(D_l^{\text{dq}}, T_l^{\text{eq}})$ by substituting the continuation value in (4.23) with the approximated continuation value:

$$\widehat{V}_l(D_l^{\text{dq}}, T_l^{\text{eq}}) = \max \left\{ U_l^{\text{lt}}(D_l^{\text{dq}}, T_l^{\text{eq}}), \widehat{C}_\theta(l+1, D_l^{\text{dq}}, T_l^{\text{eq}}) \right\}. \quad (4.26)$$

Then, the reference continuation value can be obtained according to (4.24):

$$\tilde{C}_l(D_l^{\text{dq}}, T_l^{\text{eq}}) = \mathbb{E} \left[\widehat{V}_{l+1}(D_{l+1}^{\text{dq}}, T_{l+1}^{\text{eq}}) | \mathbf{D}_l^{\text{dq}}, \mathbf{T}_l^{\text{eq}} \right]. \quad (4.27)$$

Due to the unknown statistics, we use the single-sample estimation method [143] to approximate the expectation term in (4.27):

$$\mathbb{E} \left[\widehat{V}_{l+1}(D_{l+1}^{\text{dq}}, T_{l+1}^{\text{eq}}) | \mathbf{D}_l^{\text{dq}}, \mathbf{T}_l^{\text{eq}} \right] \approx \widehat{V}_{l+1}(D_{l+1}^{\text{dq}}, T_{l+1}^{\text{eq}}). \quad (4.28)$$

Finally, by combining (4.27) and (4.28), and calculating $\widehat{V}_{l+1}(D_{l+1}^{\text{dq}}, T_{l+1}^{\text{eq}})$ in (4.28) according to (4.26), the reference continuation value is obtained by:

$$\tilde{C}_l(D_l^{\text{dq}}, T_l^{\text{eq}}) = \max \left\{ U_{l+1}^{\text{lt}}(D_{l+1}^{\text{dq}}, T_{l+1}^{\text{eq}}), \widehat{C}_\theta(l+2, D_{l+1}^{\text{dq}}, T_{l+1}^{\text{eq}}) \right\}. \quad (4.29)$$

Remark 1: To calculate the reference continuation value $\tilde{C}_l(D_l^{\text{dq}}, T_l^{\text{eq}})$ in (4.29), D_{l+1}^{dq} and T_{l+1}^{eq} are required. As a result, with $\{D_l^{\text{dq}}\}_{l=0}^{x_n}$ and $\{T_l^{\text{eq}}\}_{l=0}^{x_n}$ collected in the offloading decision-making for the n -th task, reference continuation values before executing each of the first x_n layers of the shallow DNN can be obtained, where $x_n \leq l_e + 1$. By contrast, with DT-assisted data augmentation, which generates $\{D_l^{\text{dq}}\}_{l=x_n+1}^{l_e+1}$ and $\{T_l^{\text{eq}}\}_{l=x_n+1}^{l_e+1}$, reference continuation values before executing each of the $l_e + 1$ layers in the shallow DNN can be obtained. In this way, the data for training ContValueNet is augmented.

3) *Online ContValueNet Training*: To mitigate the impact of the one-sample estimation method in (4.28), ContValueNet is trained in an online manner with a carefully designed loss function. Specifically, the loss function for training ContValueNet after processing the

first $\hat{n}$ DNN tasks, where $1 \leq \hat{n} \leq M$, is denoted by $L^{\text{NN}}(\theta, \hat{n})$ and defined by:

$$L^{\text{NN}}(\theta, \hat{n}) = \frac{1}{\hat{n}(l_e + 1)} \sum_{n=0}^{\hat{n}} \sum_{l=0}^{l_e} \left[\hat{C}_\theta(l+1, D_l^{\text{Iq}}, T_l^{\text{eq}}) - \tilde{C}_l(D_l^{\text{Iq}}, T_l^{\text{eq}}) \right]^2, \quad (4.30)$$

which is the mean squared error of the continuation value approximation. Then, the gradient descent method is applied to optimize the parameters θ in ContValueNet to minimize the loss function. Specifically, the update rule of θ is:

$$\theta' = \theta - \gamma \nabla_\theta L^{\text{NN}}(\theta, \hat{n}), \quad (4.31)$$

where γ is the learning rate; $\nabla_\theta L^{\text{NN}}(\theta, \hat{n})$ is the first-order derivative of the loss function with respect to θ .

4.6 Offloading Decision Space Reduction

The complexity of the approach to adaptive device-edge collaboration on DNN inference in Section 4.5.1 can be high due to the potentially large number of layers in the DNN. In this section, from two aspects, we will derive necessary conditions for an offloading decision to be optimal in maximizing the expected long-term utility. Correspondingly, we investigate decision space reduction to reduce the complexity.

4.6.1 DNN Layers with Negligible Execution Time

The execution time of some layers in a DNN, e.g., pooling layers in a convolutional neural network (CNN), is negligible due to relatively simple operations [144]. As a result, such a layer has a negligible impact on the long-term on-device queuing delay and the edge queuing delay of a DNN task. By contrast, the data size before and after the execution of such layers can be drastically changed. For example, a max-pooling layer downsamples input data to learn spatially robust features for image recognition tasks, while a max-unpooling layer upsamples the input data to restore the original data size for semantic segmentation tasks.

Remark 2: If a layer in a DNN has negligible execution time and outputs data with reduced (increased) size, offloading the DNN task before (after) the on-device execution of the layer cannot be optimal for maximizing the expected long-term utility of the DNN

task. This is because offloading a DNN task after the data size is increased or before the data size is reduced would result in a higher data uploading delay. Based on this necessary condition, we can treat each max-pooling layer and its preceding layer as one logical layer and each max-unpooling layer and its succeeding layer as one logical layer in the offloading decision-making for a DNN task.

4.6.2 Properties of Workload Evolution

Based on the properties in the evolution of the on-device workload and the edge server workload over time, properties in the evolution of long-term on-device queuing delay and edge queuing delay (as the on-device inference continues) can be derived, respectively.

Property 1. *During the on-device inference for a DNN task, the on-device workload $Q^D(t)$ is non-decreasing with t . In addition, the on-device inference delay T_n^{lc} increases with x_n . As a result, when x_n increases, the long-term on-device queuing delay $D_n^{\text{lc}}(x_n)$, calculated by (4.17), is non-decreasing with x_n . The increase of $D_n^{\text{lc}}(x_n)$ is minimized if no DNN tasks are generated after the controller starts to make the offloading decision for the n -th DNN task, i.e., $Q^D(t) = Q^D(t_{n,\hat{x}_n}), \forall t \geq t_{n,\hat{x}_n}$. The minimum increase is equal to the product of $Q^D(t_{n,\hat{x}_n})$ and the increase of on-device inference delay $T_n^{\text{lc}}(x_n)$.*

Property 2. *The edge server workload can decrease or increase during the extended on-device inference for the n -th task due to the increase of x_n . The decrease of the workload is maximized if no tasks arrive at the edge server during the extended on-device inference, and the maximum decrease is equal to the workload that the edge server can process during the extended on-device inference. As a result, given the increase of x_n , the maximum decrease of the edge queuing delay $T_n^{\text{eq}}(x_n)$, calculated by dividing the edge server workload by the edge server processing capability, equals to the increase of the on-device inference delay $T_n^{\text{lc}}(x_n)$.*

Using the above properties, a necessary condition for an offloading decision $x_n \in \{\hat{x}_n, \dots, l_e\}$ to be optimal can be derived.

Lemma 1. *Define a deterministic part in the long-term utility of the n -th DNN task as $U_n^{\text{pt}}(x_n) \stackrel{\text{def}}{=} -T_n^{\text{up}}(x_n) - T_n^{\text{ec}}(x_n) - \beta E_n(x_n)$. If $x_n^* \leq l_e$ maximizes $\mathbb{E}[U_n^{\text{lt}}(x_n)]$, then for any $x_n \in \{\hat{x}_n, \dots, x_n^*\}$, it holds:*

$$U_n^{\text{pt}}(x_n^*) \geq U_n^{\text{pt}}(x_n) + Q^D(t=t_{n,\hat{x}_n})(T_n^{\text{lc}}(x_n^*) - T_n^{\text{lc}}(x_n)), \quad (4.32)$$

where $t_{n,\hat{x}_n}$ is the index of the time slot from which the n -th DNN task can be offloaded to the edge server.

Proof. Note that $U_n^{\text{lt}}(x_n) = U_n^{\text{pt}}(x_n) - T_n^{\text{lc}}(x_n) - D_n^{\text{lc}}(x_n) - T_n^{\text{eq}}(x_n) + \alpha A_n(x_n)$. If $x_n^* \leq l_e$ maximizes $\mathbb{E}[U_n^{\text{lt}}(x_n)]$, then for any $x_n \in \{\hat{x}_n, \dots, x_n^*\}$, we have:

$$\begin{aligned} & U_n^{\text{pt}}(x_n^*) - T_n^{\text{lc}}(x_n^*) - \mathbb{E}[D_n^{\text{lc}}(x_n^*)] - \mathbb{E}[T_n^{\text{eq}}(x_n^*)] + \alpha A_n(x_n^*) \\ & \geq U_n^{\text{pt}}(x_n) - T_n^{\text{lc}}(x_n) - \mathbb{E}[D_n^{\text{lc}}(x_n)] - \mathbb{E}[T_n^{\text{eq}}(x_n)] + \alpha A_n(x_n). \end{aligned} \quad (4.33)$$

When $x_n \leq l_e$ (including the case when $x_n^* \leq l_e$), the n -th DNN task is offloaded to the edge server for processing by the full-size DNN. The inference accuracy is thus the same under the two cases:

$$A_n(x_n^*) = A_n(x_n). \quad (4.34)$$

According to *Property 1*, we have:

$$D_n^{\text{lc}}(x_n^*) \geq D_n^{\text{lc}}(x_n) + Q^D(t = t_{n, \hat{x}_n})(T_n^{\text{lc}}(x_n^*) - T_n^{\text{lc}}(x_n)). \quad (4.35)$$

According to *Property 2*, we have:

$$T_n^{\text{lc}}(x_n^*) - T_n^{\text{lc}}(x_n) \geq T_n^{\text{eq}}(x_n) - T_n^{\text{eq}}(x_n^*). \quad (4.36)$$

Combining (4.33), (4.34), (4.35) and (4.36), (4.32) is obtained. $\square$

In addition, we derive a necessary condition of the device-only inference, i.e., $x_n = l_e + 1$, to be optimal in maximizing the *long-term utility* $U_n^{\text{lt}}(x_n)$.⁴

Lemma 2. *If device-only inference, i.e., $x_n = l_e + 1$, maximizes the long-term task utility $U_n^{\text{lt}}(x_n)$, then, $U_n(l_e + 1)$, i.e., the task utility under the device-only inference, must satisfy the following condition:*

$$U_n(l_e + 1) \geq U_n(\hat{x}_n) + Q^D(t = t_{n, \hat{x}_n})(T_n^{\text{lc}}(l_e + 1) - T_n^{\text{lc}}(\hat{x}_n)). \quad (4.37)$$

Proof. Since $x_n = l_e + 1$ maximizes the long-term task utility $U_n^{\text{lt}}(x_n)$, we have:

$$U_n^{\text{lt}}(l_e + 1) \geq U_n^{\text{lt}}(\hat{x}_n). \quad (4.38)$$

⁴The long-term utility of a task by device-only inference would not be affected by the dynamic workload at the edge server, while the utility by the other inference scenarios would be affected. Due to the unknown statistics of the dynamic workload, we cannot derive any necessary condition for device-only inference to be optimal in maximizing the *expected* long-term utility.

According to *Property 1*, we have:

$$D_n^{\text{lc}}(l_e + 1) \geq D_n^{\text{lc}}(\hat{x}_n) + Q^D(t = t_{n, \hat{x}_n})(T_n^{\text{lc}}(l_e + 1) - T_n^{\text{lc}}(\hat{x}_n)). \quad (4.39)$$

Note that:

$$U_n^{\text{lt}}(l_e + 1) + D_n^{\text{lc}}(l_e + 1) = U_n(l_e + 1) + T_n^{\text{lc}}(\mathbf{x}_{n-1}), \quad (4.40a)$$

$$U_n^{\text{lt}}(\hat{x}_n) + D_n^{\text{lc}}(\hat{x}_n) = U_n(\hat{x}_n) + T_n^{\text{lc}}(\mathbf{x}_{n-1}). \quad (4.40b)$$

Combining (4.38) and (4.39), (4.37) is obtained. $\square$

Based on *Lemma 1* and *Lemma 2*, an algorithm for reducing the offloading decision space is proposed in Algorithm 1. Specifically, when the n -th DNN task can be offloaded to the edge server, the potential offloading decisions $x_n \in \{\hat{x}_n, \hat{x}_n + 1, \dots, l_e\}$ will be checked, and only the decisions satisfying the condition given in *Lemma 1* will be considered in the learning-assisted offloading decision-making. In addition, if all the possible offloading decisions $x_n \in \{\hat{x}_n + 1, \dots, l_e\}$ violate the necessary condition given in *Lemma 1*, the condition in *Lemma 2* will be checked to determine whether device-only inference, i.e., $x_n = l_e + 1$, can be optimal for maximizing the long-term task utility. If not, $x_n = l_e + 1$ will not be considered. Finally, only the remaining decisions, i.e, $\mathcal{L}_n$, will be considered.

Algorithm 1: Decision Space Reduction

```

1 Input:  $\hat{x}_n$ ;
2 Output:  $\mathcal{L}_n$ ;
3 Initialization:  $\mathcal{L}_n = \{\hat{x}_n, \hat{x}_n + 1, \dots, l_e + 1\}$ ;
4 for  $l \in \{\hat{x}_n, \hat{x}_n + 1, \dots, l_e\}$  do
5   |   Given  $x_n^* = l$ , for any  $x_n \in \{\hat{x}_n, \dots, x_n^*\}$ , if (4.32) is violated, delete  $l$  in the set
   |    $\mathcal{L}_n$ ;
6 end
7 if  $\mathcal{L}_n = \{\hat{x}_n, l_e + 1\}$  then
8   |   if (4.37) is violated then
9     |   delete  $l_e + 1$  in the set  $\mathcal{L}_n$ ;
10  |   end
11 end
12 return  $\mathcal{L}_n$ 

```

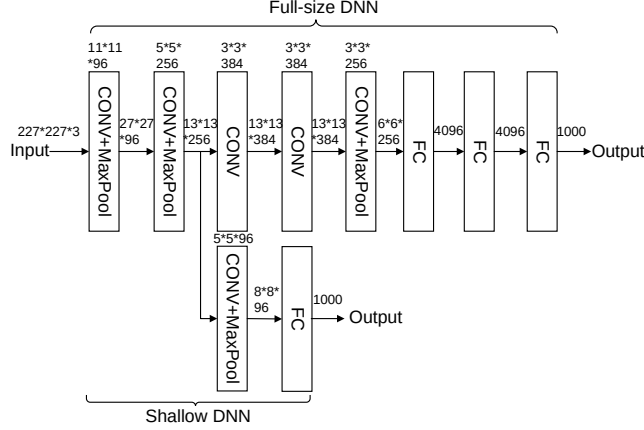


Fig. 4.6: Full-size DNN and shallow DNN.

4.7 Simulation Results

In this section, through simulations, we evaluate the performance of the DT-assisted approach to adaptive device-edge collaboration on DNN inference.

4.7.1 Simulation Settings and Benchmarks

The main parameters used in the simulations are given in Table 4.1, and the detailed configurations of the full-size and shallow DNNs are shown in Fig. 4.6, where Alexnet is chosen as the full-size DNN. Based on *Remark 2*, we consider each pooling layer and its preceding layer as one logical layer. The execution time for the layers of the shallow DNN by the device, i.e., $\{d_l^D\}_{l=1}^{l_e+1}$, and that for the layers of the full-size DNN by the edge server, i.e., $\{d_l^E\}_{l=1}^L$, are estimated based on the number of FLOPs for the layers and the computation frequency of the AIoT device and the edge server[139].

The neural network for approximating the continuation values, i.e., ContValueNet, consists of three fully-connected layers with 200, 100 and 20 neurons, respectively. For optimizing the parameters therein, the learning rate γ is set to be 1×10^{-3} and the Adam optimizer is chosen. In the processing of the first 2000 DNN tasks, ContValueNet for continuation value approximation is trained, i.e., $M = 2000$. The average DNN task utility, inference delay, accuracy and energy consumption are derived by applying the trained ContValueNet to assist making offloading decisions of 8000 DNN tasks. The DNN task generation at the considered device follows a Bernoulli distribution with probability

p and task arrival from the other devices to the edge server follows a Poisson distribution with arrival rate λ . The CPU cycles required for processing a computing task follows uniform distribution $U(0, U^{\max})$, with the value of $U^{\max}$ set as 8×10^9 . The DNN task generation rate in the unit of tasks per second can be calculated by p/Δ . In addition, the edge processing load can be calculated by $\lambda U^{\max}/2f^E$. As a unitless ratio, the edge processing load is not tied to a specific system parameter such as computing capability, task arrival rate, and task computation cost. As a result, using this metric allows our simulation results to offer findings that are applicable across various AIoT network settings.

Table 4.1: Main simulation settings in DNN-based sensing data processing

Parameters	Symbol	Value
Time duration of a time slot	ΔT	10 ms
Exit layer index	l_e	2
Computation frequency of the edge server	f^E	50 GHz
Computation frequency of the AIoT device	f^D	1 GHz
Accuracy of full-size DNN	η^E	0.9
Accuracy of shallow DNN	η^D	0.6
Uplink transmission rate	R_0	126 Mbps
Transmit power of the AIoT device	p^{up}	20 dBm
Energy coefficient	κ^E, κ^D	10^{-30}
Weight for accuracy	α	1.0
Weight for energy consumption	β	0.2

In the simulations, the following three benchmarks are used:

- **One-Time Ideal Case:** The offloading decision of a DNN task is made only once (upon task generation) to maximize the long-term utility of the task, assuming perfect knowledge of the future on-device workload and edge server workload.
- **One-Time Long-term:** The offloading decision of a DNN task is made only once (upon task generation) to maximize the long-term utility of the DNN task based on the current on-device workload and edge server workload.
- **One-Time Greedy:** Similar to [32], the offloading decision of a DNN task is made only once (upon task generation) to maximize the utility of the DNN task based on the current on-device workload and edge server workload.

In addition, we will compare the performance of the learning-assisted task offloading algorithm *i*) with and without the DT-assisted training data augmentation; *ii*) with and without the offloading decision space reduction, respectively.

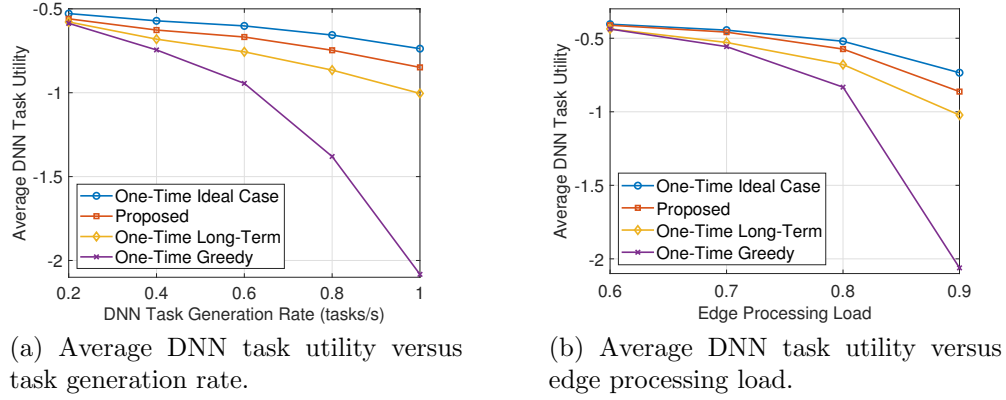


Fig. 4.7: (a) Average DNN task utility versus task generation rate. (b) Average DNN task utility versus edge processing load.

4.7.2 Adaptiveness to Dynamic Computing Workload

In Fig. 4.7(a), the average utility of DNN tasks under varying DNN task generation rate is shown, where the edge processing load is 0.9. It can be observed that the proposed approach outperforms the one-time greedy benchmark. This is because the proposed approach makes the offloading decision of a DNN task while considering the on-device queuing delay of subsequent DNN tasks. In addition, the proposed approach outperforms the one-time long-term benchmark, since instead of making task offloading decision upon the task generation, our approach continuously monitors the on-device workload during the on-device inference for each task and adaptively makes the offloading decision.

In Fig. 4.7(a), it can be observed that the performance gain compared to the one-time long-term benchmark increases with DNN task generation rate. This is because the increase of the task generation rate results from the increase of the task generation probability, which in turn increases the variance of the Bernoulli distribution and the dynamics of the on-device workload. Since the performance gain comes from better adaptiveness to the dynamics, it increases with the dynamics. In Fig. 4.7(b), the average DNN task utility under varying edge processing load is shown, where the DNN task generation rate is 1.0. Similarly, the proposed approach outperforms the one-time long-term and one-time greedy benchmarks and the performance gain increases with the edge processing load.

In Fig. 4.8, the average inference delay, accuracy and energy consumption under varying DNN task generation rate are shown, where the edge processing load is 0.9. Compared with the benchmarks, the proposed approach achieves lower delay and higher inference

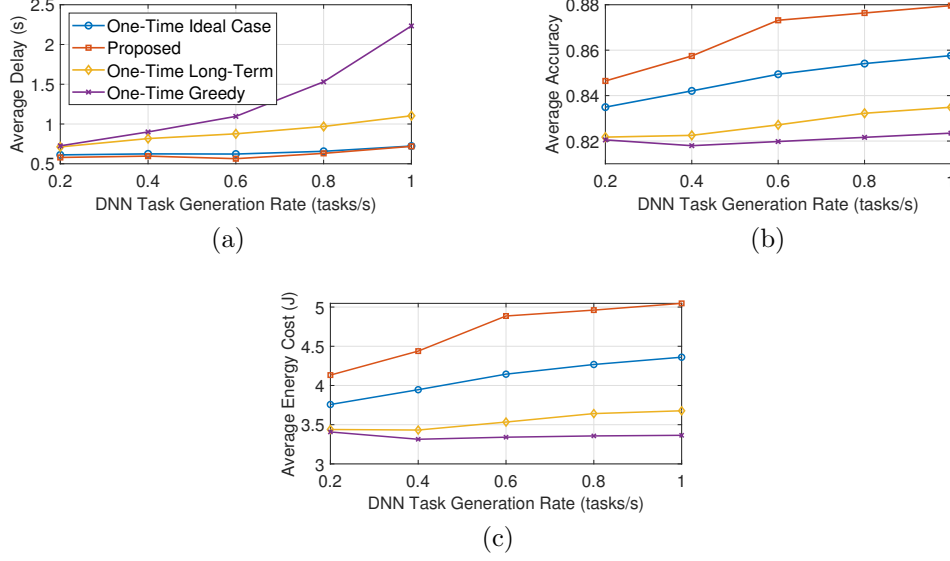


Fig. 4.8: (a) Average delay; (b) Average accuracy; (c) Average energy consumption versus DNN task generation rate.

accuracy at the cost of higher energy consumption. This results from the weights for the delay, inference accuracy and the energy consumption in the task utility, which are set as 1.0, 1.0, and 0.002, respectively. Combining the ranges of the three metrics shown in Fig. 4.8, it can be seen that the delay and the accuracy contribute significantly more than the energy consumption in determining the task utility. Since the proposed approach does not assume perfect knowledge of the workload evolution, it can conservatively offload a task to the edge server for reducing the on-device queuing delay of the tasks in the on-device queue and increasing the inference accuracy.

4.7.3 Effectiveness of DT-Assisted Training Data Augmentation

In Fig. 4.9, the number of training samples collected during the training of ContValueNet is shown, where the edge processing load is 0.9, and the DNN task generation rate is 0.4 and 0.8 in Fig. 4.9(a) and Fig. 4.9(b), respectively. It can be observed that with DT-assisted training data augmentation, the number of training samples linearly grows as the training continues. This is because the DT of workload evolution is used by the network controller to emulate the on-device and edge server workload under every possible offloading scenario so that, for each DNN task, we can obtain $l_e + 1$ training samples, where l_e is 2 based

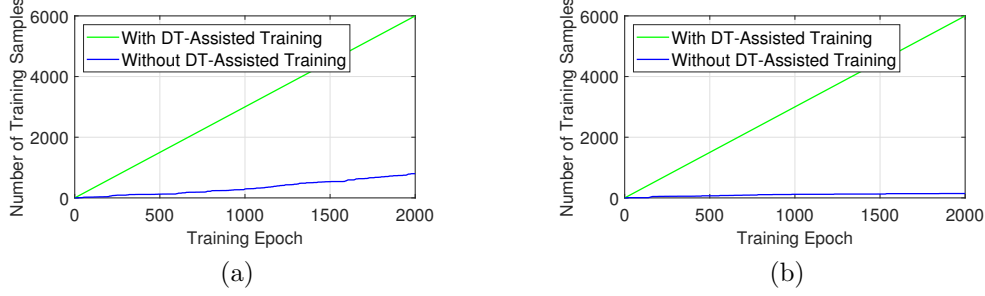


Fig. 4.9: Number of collected training samples with and without DT-assisted training data augmentation.

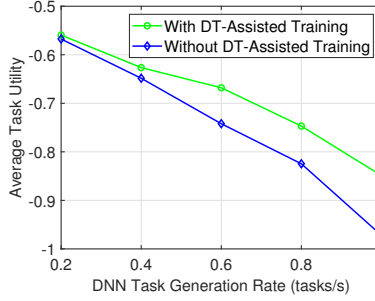


Fig. 4.10: Average DNN task utility with and without DT-assisted training data augmentation.

on the configuration of the shallow DNN in Fig. 4.6. In contrast, without DT-assisted training data augmentation, very limited training samples can be obtained since only the workload evolution before the offloading of a DNN task can be used to construct training samples.

In Fig. 4.10, the average DNN task utility versus the task generation rate, with and without DT-assisted training data augmentation, is shown, where the edge processing load is 0.9. With DT-assisted training data augmentation, the average task utility increases. In addition, as the task generation rate increases, the performance gain increases. This is because the dynamics of the on-device workload increase with the task generation rate, which in turn requires a larger amount of and more diverse training data. The performance gain due to the DT-assisted data augmentation can also be explained by analyzing the training loss during the training of ContValueNet. In Fig. 4.11, with the same simulation settings as those used to generate Fig. 4.9, we can observe that without DT-assisted data

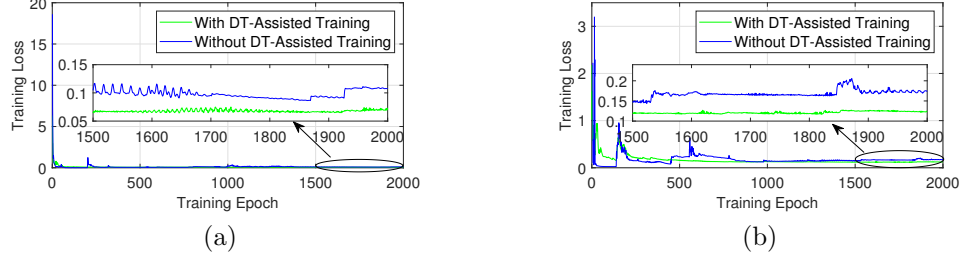


Fig. 4.11: Training loss with and without DT-assisted training data augmentation.

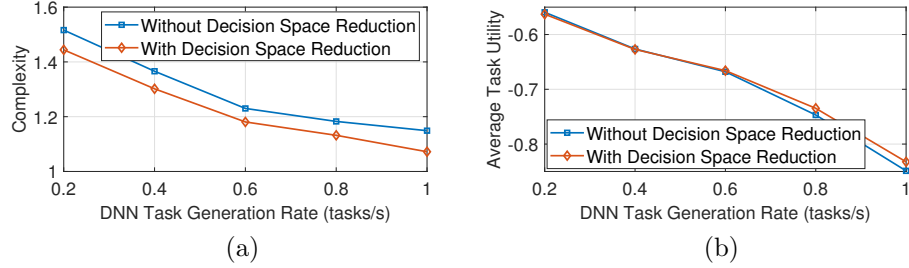


Fig. 4.12: (a) Complexity and (b) Average DNN task utility with and without decision space reduction.

augmentation, the training loss fluctuates more due to overfitting caused by insufficient training data. In contrast, with DT-assisted data augmentation, the training loss decreases more steadily to a lower level at the end of the training.

4.7.4 Complexity Reduction by Decision Space Reduction

In Fig. 4.12, the performance of the proposed device-edge collaborative DNN inference with and without decision space reduction in Algorithm 1 is shown, where the edge processing load is 0.9. As shown in Fig. 4.12(a), with the decision space reduction, the proposed approach can be implemented with reduced complexity in terms of the average number of times when the controller determines whether to continue on-device inference for a DNN task based on (4.25). This is because with the decision space reduction, the on-device DNN inference continues if the corresponding offloading decision violates the necessary condition given in *Lemma 1*. In addition, in Fig. 4.12(b), it can be observed that with the decision space reduction, the average task utility is nearly unaffected and even improved in the high

regime of DNN task generation rate. This is because the necessary conditions sometimes prevent the controller from choosing non-optimal offloading decisions that could have been chosen due to the imperfect continuation value approximation by ContValueNet.

4.8 Conclusion

In this paper, we have proposed a DT-assisted approach to device-edge collaboration on DNN inference, which can adapt to dynamic computing workloads at AIoT devices and edge servers with low signaling overhead. The proposed approach can be applied to support AIoT scenarios where densely deployed AIoT devices dynamically generate AI model inference tasks.

Chapter 5

Resource Allocation in Sensing-Assisted AR Streaming

In this chapter, we present a novel model-assisted learning scheme for resource allocation in sensing-assisted mobile AR streaming. In our considered scenario, sensing is performed by the wireless network infrastructure for user positioning and environment mapping to assist in resource-efficient content visibility-aware virtual content delivery. The goal is to minimize the long-term radio resource consumption for mobile AR streaming by optimizing the radio resource allocation for user positioning and environment mapping. In specific, we first develop a mathematical model to estimate the content visibility uncertainty and the content delivery resource consumption. We then generate a reference resource allocation decision that guides a DRL-based decision process to efficiently adapt to non-stationary user and environment dynamics. Trace-driven simulations demonstrate that, the proposed scheme significantly reduces radio resource consumption for delivering virtual content visible to an individual AR user, compared to benchmark schemes.

5.1 Introduction

As a key form of immersive communications in future 6G wireless networks, mobile AR enables users to move freely and view virtual content overlaid on their surrounding physical environment [18, 110, 145]. Due to the large data size of high-fidelity virtual content and limited FoV of AR users, it is resource-efficient to deliver only the portion of the virtual content that is visible to a user according to the real-time user position [10, 146]. However,

the virtual content delivery is challenged by the high uncertainty of user positioning in certain cases, such as low-light conditions, rapid user motion, and NLoS scenarios.

Environment mapping by wireless devices and infrastructures under the ISAC paradigm is receiving increasing attention in enhancing communication service provision [147]. Particularly, given an environment map that contains the positions of environment elements such as reflectors and scatters, the NLoS radio signal paths can be characterized to enhance user positioning [148]. As a result, the uncertainty in virtual content visibility can be reduced, which is beneficial to resource-efficient content delivery. Nevertheless, balancing resource consumption among environment mapping, user positioning, and virtual content delivery is challenging. This is because the user’s surrounding environment changes dynamically due to user mobility, and the benefit provided by the current environment for resource-efficient content delivery also varies with time. The temporal non-stationarity poses challenges in efficiently learning the optimal resource allocation policy [149].

In this chapter, we investigate the resource allocation for environment-aware content delivery in mobile AR. We aim to minimize the long-term radio resource consumption for delivering virtual content visible to an individual AR user by optimizing the radio resource allocation for user positioning and environment mapping. To this end, we first develop a mathematical model to estimate the uncertainty in virtual content visibility and the resource consumption for virtual content delivery. Second, we propose a model-assisted learning scheme to efficiently obtain the optimal resource allocation decision-making policy under non-stationary user mobility and environment dynamics. Specifically, we generate a reference decision using the mathematical model-based optimization, and analyze the relation between the reference decision and the optimal decision to reduce the action space of DRL-based decision making. Simulation experiments using real-world AR user traces demonstrate that the proposed scheme significantly reduces resource consumption compared to benchmark schemes.

5.2 System Model

5.2.1 AR Content Delivery Model

As shown in Fig. 5.1, we consider a physical area covered by an AP, where a set of virtual content items, denoted by $\mathcal{U} = \{1, 2, \dots, U\}$, are positioned in the virtual space overlaid on the physical area. Time is divided into time intervals indexed by $t \in \mathcal{T} = \{1, 2, \dots, T\}$. Each virtual content item is a volumetric video partitioned into segments, with each corresponding to a time interval. An AR user, equipped with a head-mounted AR device,

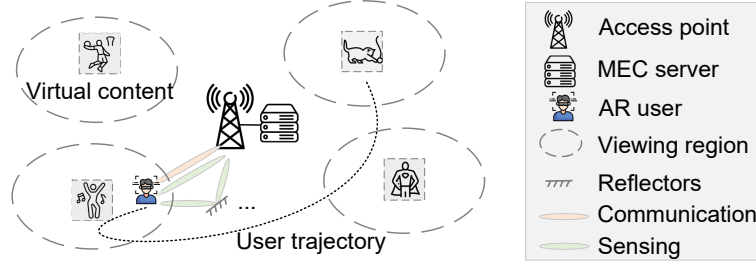


Fig. 5.1: Sensing-assisted mobile AR streaming model.

moves within the physical area in all the considered time intervals. When the user is within the viewing region of an item, the AP will deliver the corresponding volumetric video segments to the user. For simplicity, we assume that the viewing regions of different items do not overlap, and the user views the items one by one. At the beginning of the t -th time interval, the index of the virtual content item that is viewed by the user is denoted by $u(t) \in \mathcal{U} \cup \{0\}$, with $u(t) = 0$ indicating that the user is not viewing any virtual content item.

In the t -th time interval, the AP proactively delivers the video segment of the $u(t)$ -th virtual content item in the $(t + 1)$ -th time interval to the user. For communication efficiency, the AP determines and delivers only the visible portion of the video segment that probably falls into the FoV of the user. The content visibility is estimated based on the user position, denoted by $\mathbf{p}^{(t)} = [p_{t,x}, p_{t,y}]^T$, the user orientation denoted by α , and the center position of the $u(t)$ -th virtual content item, denoted by $\mathbf{o}^{u(t)} = [o_{u(t),x}, o_{u(t),y}]^T$. For simplicity, we assume that the user is oriented towards the center of the virtual content item, whose position is fixed and known. For user positioning, the downlink bi-static sensing is implemented when the on-device sensors (such as cameras) are error-prone, indicated by $\eta^{(t)} = 1$, and otherwise, we have $\eta^{(t)} = 0$. The indicator $\eta^{(t)}$ is considered as a random variable, whose value is revealed when the t -th time interval begins.

5.2.2 Sensing Model

In this system, the bi-static sensing mode is employed for user positioning and environment mapping. Specifically, the AP transmits signals that are received by the AR device and used to extract information about the user's position and environment.

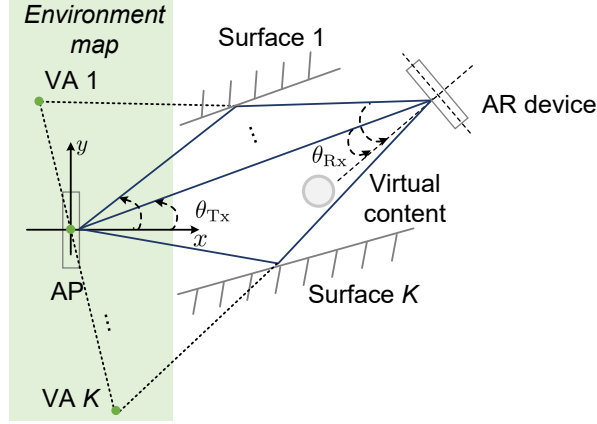


Fig. 5.2: Signal propagation and environment map models.

Propagation Model

As shown in Fig. 5.2, for the signal propagation from the AP to the AR device, we consider the line-of-sight (LoS) path, and the NLoS paths where signals are reflected by smooth surfaces and then arrive at the AR device. Signals towards the AR device within the viewing region of the u -th virtual content item are considered to be reflected by the same set of smooth surfaces, whose number is denoted by $K^{(u)}$. Let $k \in \mathcal{K}^{(u)} = \{0, 1, 2, \dots, K^{(u)}\}$ denote the path index in the signal propagation, where $k = 0$ represents the LoS path, and $k \in \{1, 2, \dots, K^{(u)}\}$ represents the NLoS path corresponding to the k -th smooth surface and the u -th virtual content item. Each path may be blocked within a given time interval. We denote the blockage indicator for the k -th path in the t -th time interval by $\xi_k^{(t)}$, where $\xi_k^{(t)} = 0$ to indicate that the k -th path is not blocked in the t -th time interval, and $\xi_k^{(t)} = 1$ otherwise. The indicator $\xi_k^{(t)}$ is considered as a random variable, whose value is revealed after signals are transmitted in the time interval.

Environment Map Model

As shown in Fig. 5.2, by mirroring the position of the AP across the $K^{(u)}$ smooth surfaces, we obtain $K^{(u)}$ virtual anchors (VAs). As a result, the signals that are reflected by the k -th surface can be treated as if they were emitted directly from the k -th VA to the user, even when user mobility results in time-varying reflection points on the surfaces. The location information of the AP and the VAs jointly constitute an *environment map*. The environment map corresponding to the u -th virtual content item is denoted by $\mathcal{Q}^{(u)} =$

$\{\mathbf{q}_k^{(u)}, \forall k \in \mathcal{K}^{(u)}\}$, where $\mathbf{q}_0^{(u)} = [q_{0,x}, q_{0,y}]^T$ represents the position of the AP, and $\mathbf{q}_k^{(u)} = [q_{k,x}^{(u)}, q_{k,y}^{(u)}]^T, \forall k \in \{1, 2, \dots, K^{(u)}\}$ represents the position of the k -th VA, corresponding the u -th virtual content item, which is fixed but needs to be estimated.

At the beginning of the t -th time interval, the AP has a *global* estimate of the environment map corresponding to the $u(t)$ -th virtual content item, denoted by $\bar{\mathbf{Q}}^{(u(t),t)} = \{\bar{\mathbf{q}}_k^{(u(t),t)} \mid \forall k \in \mathcal{K}^{(u(t))}\}$, which is obtained by fusing the *one-interval* estimates of the environment map in preceding time intervals. Here, $\bar{\mathbf{q}}_k^{(u(t),t)}$ represents the global estimate of $\mathbf{q}_k^{(u(t))}$ at the beginning of the t -th time interval, which has an uncertainty of $\bar{\varepsilon}_{q_k}^{(u(t),t)}$ in terms of squared positioning error. Specially, $\bar{\varepsilon}_{q_k}^{(u(t),t)} = 0$ for $k = 0$ since the location of AP, i.e., $\mathbf{q}_0$, is perfectly known. The uncertainty of the environment map, denoted by $\bar{\mathcal{E}}^{(u(t),t)}$, is defined as the set of the uncertainties in the global estimates of all the VAs, i.e., $\bar{\mathcal{E}}^{(u(t),t)} = \{\bar{\varepsilon}_{q_k}^{(u(t),t)} \mid \forall k \in \mathcal{K}^{(u(t))}\}$. Both the global estimate and the uncertainty of the environment map at the beginning of the t -th time interval, i.e., $\bar{\mathbf{Q}}^{(u(t),t)}$ and $\bar{\mathcal{E}}^{(u(t),t)}$, can be used for user positioning in the t -th time interval, and will be updated if subsequent one-interval estimate of the environment map is obtained during this interval.

Sensing Signal Transmission

In the t -th time interval when $\eta^{(t)} = 1$, the AP sequentially transmits sensing signals lasting $m_k^{(t)}$ time slot towards the previously estimated direction of the k -th path for all $k \in \mathcal{K}^{(u(t))}$. Due to the lack of information on the real-time signal attenuation in different NLoS paths, the sensing signals over all the NLoS paths are allocated with the same number of time slots. The time slot allocation decisions are thus re-written as $m_k^{(t)} = m_{\text{LoS}}^{(t)}, k = 0$, and $m_k^{(t)} = m_{\text{NLoS}}^{(t)}$ for all $k \in \mathcal{K}^{(u(t))} \setminus \{0\}$. We consider an Orthogonal Frequency Division Multiplexing (OFDM) system, and define a radio Resource Element (RE) as the resource unit that occupies one OFDM symbol in the time domain and one subcarrier in the frequency domain. The overall number of REs allocated to the sensing signals, for user positioning and environment mapping, can be calculated as $z_{\text{sen}}^{(t)} = \sum_{k=0}^{K^{(u(t))}} m_k^{(t)} N_t N_s^{\text{sen}}$, where N_t is the number of OFDM symbols that each time slot contains, and N_s^{sen} represents the number of subcarriers that each sensing signal occupies.

5.3 Analysis and Problem Formulation

We develop a mathematical model to estimate the uncertainties associated with user positioning, environment mapping, and content visibility, as well as the radio resource con-

sumption involved in virtual content delivery. Then, we formulate the resource allocation problem.

5.3.1 User Positioning and Environment Map Uncertainties

Channel Parameter Estimation

Define a vector of the channel parameters corresponding to k -th path as $\Theta_k^{(t)} = [\tau_k^{(t)}, \theta_{R,k}^{(t)}]^T$, where $\tau_k^{(t)}$ represents the path delay and $\theta_{R,k}^{(t)}$ represents the angle of arrival. The Equivalent Fisher Information Matrix (EFIM) of the parameter vector is denoted by $\mathbf{J}_k^{(t)}$. According to [150, Equation 14], $\mathbf{J}_k^{(t)}$ is calculated as

$$\mathbf{J}_k^{(t)} = \Gamma_k^{(t)} \begin{bmatrix} \frac{2\pi^2((N_s^{\text{sen}})^2 - 1)}{3(N_s^{\text{max}})^2 T_s^2} & 0 \\ 0 & \frac{\pi^2(N_R^2 - 1) \cos^2(\theta_{R,k}^{(t)})}{6} \end{bmatrix}, \quad (5.1)$$

where N^{max} and T_s represent the total number of subcarriers and the sampling interval for an OFDM system, respectively; N_R is the number of antennas at the AR device; and $\Gamma_k^{(t)} \triangleq (1 - \xi_k^{(t)}) \text{SNR}_k^{(t)} N_R N_s^{\text{sen}} m_k^{(t)}$. Here, $\text{SNR}_k^{(t)} \triangleq P_T G_T |\gamma_k^{(t)}|^2 / (N_0 B)$, where P_T is the transmit power of the AP; G_T is the transmit array gain; $\gamma_k^{(t)}$ is the complex channel gain for the k -th path in the t -th time interval; N_0 is the one-side noise power spectral density (PSD); B is the total bandwidth of the OFDM system. According to the information inequality, the Cramér-Rao lower bounds of the estimation errors for $\tau_k^{(t)}$ and $\theta_{R,k}^{(t)}$, denoted by $\tilde{\epsilon}_{\tau_k}^{(t)}$ and $\tilde{\epsilon}_{\theta_{R,k}}^{(t)}$, are calculated as $\tilde{\epsilon}_{\tau_k}^{(t)} = [\mathbf{J}_k^{(t)}]_{11}^{-1}$ and $\tilde{\epsilon}_{\theta_{R,k}}^{(t)} = [\mathbf{J}_k^{(t)}]_{22}^{-1}$.

AR User Positioning

The channel parameters corresponding to the k -th path, $\forall k \in \mathcal{K}^{(t)}$, can be determined as $\tau_k^{(t)} = \|\mathbf{p}^{(t)} - \mathbf{q}_k^{(u(t))}\|_2 / c$; and $\theta_{R,k}^{(t)} = \pi + \alpha - \arccos((p_{t,x} - q_{k,x}^{(u(t))}) / \|\mathbf{p}^{(t)} - \mathbf{q}_k^{(u(t))}\|_2)$. Here, c is the speed of light. Accordingly, leveraging the estimates of channel parameters, i.e., $\tau_k^{(t)}$ and $\theta_{R,k}^{(t)}$, and the estimates of the VAs' positions, i.e., $\mathbf{q}_k^{(u(t))}$, the user position $\mathbf{p}^{(t)}$ can be estimated as $\hat{\mathbf{p}}^{(t)}$.

Proposition 7. *The uncertainty of the user position estimate $\hat{\mathbf{p}}^{(t)}$, in terms of the average of the squared positioning error bounds along the two coordinate axes, is denoted by $\tilde{\epsilon}_p^{(t)}$*

and calculated as:

$$\tilde{\varepsilon}_p^{(t)} = \text{tr}((\sum_{k=0}^{K^{u(t)}} \lambda_k^{(t)} (\mathbf{v}_k^{(t)})^T \mathbf{v}_k^{(t)} + \mu_k^{(t)} (\mathbf{w}_k^{(t)})^T \mathbf{w}_k^{(t)})^{-1})/2, \quad (5.2)$$

where we have:

$$\lambda_k^{(t)} = 1 / (c^2 \cdot \check{\varepsilon}_{\tau_k}^{(t)} + \bar{\varepsilon}_{q_k}^{(u(t),t)}); \quad (5.3a)$$

$$\mu_k^{(t)} = 1 / \left(\|\mathbf{p}^{(t)} - \mathbf{q}_k^{(u(t))}\|_2^2 \cdot \check{\varepsilon}_{\theta_{T,k}}^{(t)} + \bar{\varepsilon}_{q_k}^{(u(t),t)} \right); \quad (5.3b)$$

$$\mathbf{v}_k^{(t)} = [\cos \theta_{T,k}^{(t)}, \sin \theta_{T,k}^{(t)}]^T; \quad (5.3c)$$

$$\mathbf{w}_k^{(t)} = [-\sin \theta_{T,k}^{(t)}, \cos \theta_{T,k}^{(t)}]^T. \quad (5.3d)$$

Here, $\theta_{T,k}^{(t)} = \pi + \alpha - \theta_{R,k}^{(t)}$ represents the angle of departure.

Proof. Please refer to Appendix F. □

Environment Map Update

Given the channel parameter estimation, the AR device obtains new one-interval position estimates of all the VAs, denoted by $\{\hat{\mathbf{q}}_k^{(u(t),t)} \mid \forall k \in \mathcal{K}^{u(t)}\}$, which is then used to update the global environment map estimate. For tractability, we use only the estimate of the channel parameters corresponding the 0-th path and the k -th path to approximate the uncertainty of $\hat{\mathbf{q}}_k^{(u(t),t)}$.

Proposition 8. *The uncertainty of $\hat{\mathbf{q}}_k^{(u(t),t)}$, denoted by $\check{\varepsilon}_{q_k}^{(u(t),t)}$, can be approximated as*

$$\check{\varepsilon}_{q_k}^{(u(t),t)} = (\sum_{k' \in \{0,k\}} c^2 \cdot \check{\varepsilon}_{\tau_{k'}}^{(t)} + \|\mathbf{p}^{(t)} - \mathbf{q}_{k'}^{(u(t))}\|_2^2 \cdot \check{\varepsilon}_{\theta_{R,k'}}^{(t)})/2. \quad (5.4)$$

Proof. Please refer to Appendix G. □

Based on the inverse-variance weighting principle, the global estimate of the k -th VA's position can be updated as $\bar{\mathbf{q}}_k^{(u(t),t+1)}$, and its uncertainty is updated as

$$\bar{\varepsilon}_{q_k}^{(u(t),t+1)} = \left(\sum_{t'=0}^t \mathbf{1}_{\{u(t')=u(t)\}} \eta^{(t')} (\check{\varepsilon}_{q_k}^{(u(t'),t')})^{-1} \right)^{-1}, \quad (5.5)$$

where $\mathbf{1}_{\{u(t')=u(t)\}}$ is an indicator function representing whether the virtual content item that is viewed in the t' -th interval is the same as that in the current t -th interval; $\check{\varepsilon}_{q_k}^{(u(t),0)}$

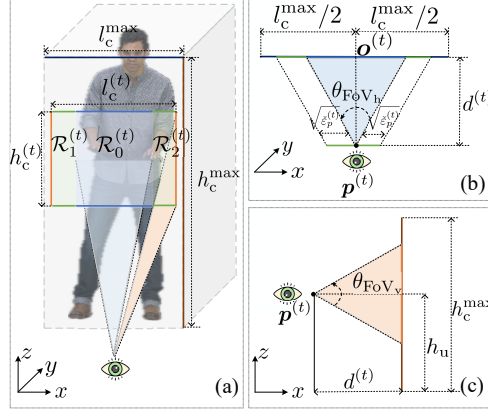


Fig. 5.3: Content visibility estimation.

represents the uncertainty of the initial estimate of the k -th VA's position by beam sweeping when the AR user enters the viewing region of the $u(t)$ -th virtual content item.

5.3.2 Content Visibility Uncertainty

Given the estimate of the AR user's position and its associated uncertainty, the AP estimates the virtual content visibility and the uncertainty of the visibility.

As shown in Fig. 5.3(a), each virtual content item is enclosed in a cube with height $h_c^{\max}$ and length $l_c^{\max}$ in meters. The region of the probably visible virtual content in the t -th time interval is denoted by $\mathcal{R}^{(t)} = \mathcal{R}_0^{(t)} \cup \mathcal{R}_1^{(t)} \cup \mathcal{R}_2^{(t)}$, where $\mathcal{R}_0^{(t)}$ represents the region that is visible to the AR user according to the position estimate of the AR user, which is shown in black in Fig. 5.3(a); $\mathcal{R}_1^{(t)}$ and $\mathcal{R}_2^{(t)}$ represent the additional regions prepared for the visibility uncertainty due to the user positioning uncertainty, which is shown in green in Fig. 5.3(a). Given the geometry relation shown in Fig. 5.3(b) and Fig. 5.3(c), the height and width of the region $\mathcal{R}^{(t)}$, denoted by $h^{(t)}$ and $l^{(t)}$, respectively, can be calculated as

$$h_c^{(t)} = \min\{h_u, d^{(t)} \cdot \tan(\frac{\theta_{\text{FoV}_v}}{2})\} + \min\{h_c^{\max} - h_u, d^{(t)} \cdot \tan(\frac{\theta_{\text{FoV}_v}}{2})\}, \quad (5.6)$$

$$l_c^{(t)} = 2 \cdot \min\{\frac{l_c^{\max}}{2}, d^{(t)} \cdot \tan(\frac{\theta_{\text{FoV}_h}}{2}) + \sqrt{\varepsilon_p^{(t)}}\}, \quad (5.7)$$

where θ_{FoV_v} and θ_{FoV_h} represent the vertical and horizontal FoVs of the AR device, respectively; h_u represents the height of the AR device; and $d^{(t)}$ represents the distance

between the position estimate of the AR user to the virtual content, which is calculated as $d^{(t)} = \|\hat{\mathbf{p}}^{(t)} - \mathbf{o}^{(u(t))}\|_2$.

5.3.3 Content Delivery Resource Consumption

The area of the probably visible region of the virtual content, denoted by $|\mathcal{R}^{(t)}|$, is calculated as $|\mathcal{R}^{(t)}| = h_c^{(t)} l_c^{(t)}$. Denote by D_0 the average data size of each virtual content item per area and time interval. The data size of the content to be delivered to the AR user in the t -th time interval is denoted by $D^{(t)}$ and calculated as $D^{(t)} = |\mathcal{R}^{(t)}| D_0$. In addition, the average transmission rate (in bits per second) of each RE in the t -th time interval is denoted by $R^{(t)}$. The number of REs required for virtual content delivery in the t -th time interval, denoted by $z_{\text{com}}^{(t)}$, is calculated as $z_{\text{com}}^{(t)} = \left\lceil \frac{D^{(t)}}{N_s^{\max} T_s R^{(t)}} \right\rceil$, where $N_s^{\max} T_s$ is the time duration of each RE.

5.3.4 Problem Formulation

The radio resource consumption in the t -th time interval for the AR user, denoted by $z_{\text{tot}}^{(t)}$, is calculated as $z_{\text{tot}}^{(t)} = z_{\text{sen}}^{(t)} + z_{\text{com}}^{(t)}$. Our objective is to minimize the overall radio resource consumption in all the considered time intervals, denoted by $\mathcal{T}' = \{t | t \in \mathcal{T}, \eta^{(t)} = 1, u(t) \in \mathcal{U}\}$, for the AR user through the decision making on the slot allocation for sensing, denoted by $\mathcal{M} = \{m_{\text{LoS}}^{(t)}, m_{\text{NLoS}}^{(t)} | \forall t \in \mathcal{T}'\}$. An optimization problem is formulated as

$$\mathbf{P}_1 : \min_{\mathcal{M}} \sum_{t \in \mathcal{T}'} z_{\text{tot}}^{(t)} \quad (5.8a)$$

$$\text{s.t.} \quad m_{\text{LoS}}^{(t)} \in \{0, 1, 2, \dots, M_{\text{LoS}}\}, \quad (5.8b)$$

$$m_{\text{NLoS}}^{(t)} \in \{0, 1, 2, \dots, M_{\text{NLoS}}\}, \quad (5.8c)$$

where M_{LoS} and M_{NLoS} are the maximum number of time slots for sensing towards the previously estimated direction of the LoS path and each NLoS path in a time interval, respectively.

We reformulate the original problem $\mathbf{P}_1$ as a Markov Decision Process (MDP) problem as follows.

- *State*: The state at the beginning of the t -th time interval, denoted by $\mathbf{s}^{(t)}$, incorporates the state of the user $\mathbf{s}_u^{(t)}$ and the state of the environment $\mathbf{s}_m^{(t)}$. Specifically, $\mathbf{s}_u^{(t)}$

includes the estimate and uncertainty of the AR user's position in the $(t-1)$ -th time interval, $\tilde{\varepsilon}_p^{(t-1)}$, i.e., $\mathbf{s}_u^{(t)} = \{\hat{\mathbf{p}}^{(t-1)}, \tilde{\varepsilon}_p^{(t-1)}\}$; and $\mathbf{s}_m^{(t)}$ includes the global estimate and the uncertainty of the $u(t)$ -th environment map, i.e., $\mathbf{s}_m^{(t)} = \bar{\mathcal{Q}}^{(u(t), t-1)} \cup \bar{\mathcal{E}}^{(u(t), t-1)}$.

- *Action*: The action, denoted by $\mathbf{a}^{(t)} = \{m_{\text{LoS}}^{(t)}, m_{\text{NLoS}}^{(t)}\}$, is the slot allocation for sensing in the t -th time interval.
- *State Transition*: The index of the next time interval, when sensing-based user positioning is required, is denoted by t' . The probability of the next state $\mathbf{s}^{(t')}$ given the current state $\mathbf{s}^{(t)}$ and the action $\mathbf{a}^{(t)}$ is denoted by $P(\mathbf{s}^{(t')}|\mathbf{s}^{(t)}, \mathbf{a}^{(t)})$.
- *Reward*: The reward function in the t -th time interval, denoted by $r^{(t)}$, is the negative number of REs consumed for the AR user in this time interval, i.e., $r^{(t)} = -z_{\text{tot}}^{(t)}$.

The MDP problem can be solved using learning-based approaches, e.g., DRL, which find an optimal decision-making policy through a trial-and-error manner without mathematically modeling the state transition probability or the reward function. However, since the transition probability distribution can be non-stationary due to time-varying user mobility pattern, promptly finding the optimal policy is challenging.

5.4 Model-Assisted Learning for Resource Allocation

In this section, we generate a reference decision that maximizes the reward in each time interval, whose mathematical estimation model is obtained in Section 5.2. By analyzing the relation of the reference decision and the optimal decision, we reduce the action space of the MDP problem. After that, we introduce the DRL algorithm that is adopted to solve the MDP problem with the reduced action space.

5.4.1 Action Space Reduction

By increasing the value of $m_{\text{NLoS}}^{(t)}$, i.e., allocating more time slots for sensing along the NLoS paths, we can reduce the uncertainty in the global estimate of the environment map. Further, the uncertainty of user positioning in any time interval, i.e., $\tilde{\varepsilon}_p^{(t')}$, is non-increasing, and the resource consumption for AR content delivery, i.e., $z_{\text{com}}^{(t')}$, $\forall t' \in \mathcal{T} \setminus \{t\}$, is also non-increasing. As a result, we have:

$$\partial z_{\text{com}}^{(t')}/\partial m_{\text{NLoS}}^{(t)} \leq 0, \forall t' \in \mathcal{T} \setminus \{t\}. \quad (5.9)$$

Based on (5.9), it can be proven that given all the other slot allocation decisions in all the considered time intervals, the value of $m_{\text{NLoS}}^{(t)}$ that minimizes the accumulated resource consumption is larger than the value that minimizes the resource consumption only in the t -th time interval, i.e.,

$$\underset{m_{\text{NLoS}}^{(t)}}{\operatorname{argmin}} \sum_{t' \in \mathcal{T}'} z_{\text{tot}}^{(t')} \geq \underset{m_{\text{NLoS}}^{(t)}}{\operatorname{argmin}} z_{\text{tot}}^{(t)}, \quad (5.10)$$

which can be intuitively explained by the long-term benefit of increasing the value of $m_{\text{NLoS}}^{(t)}$ in reducing the resource consumption, as characterized by (5.9).

Similar to the right-hand-side expression of (5.10), we generate one reference decision for slot allocation that minimizes the resource consumption only in the current interval, i.e.,

$$\{m_{\text{LoS}}^{(t),\text{ref}}, m_{\text{NLoS}}^{(t),\text{ref}}\} = \underset{\{m_{\text{LoS}}^{(t)}, m_{\text{NLoS}}^{(t)}\}}{\operatorname{argmin}} z_{\text{tot}}^{(t)}. \quad (5.11)$$

According to (5.10), the minimum value of $m_{\text{NLoS}}^{(t)}$ is changed to $m_{\text{NLoS}}^{(t),\text{ref}}$, i.e., $m_{\text{NLoS}}^{(t)} \in \{m_{\text{NLoS}}^{(t),\text{ref}}, m_{\text{NLoS}}^{(t),\text{ref}} + 1, \dots, M_{\text{NLoS}}\}$. Since increasing the number of slots for sensing along the NLoS paths generally reduces the need to sensing along the LoS path, the maximum value of $m_{\text{LoS}}^{(t)}$ is changed to $m_{\text{LoS}}^{(t),\text{ref}}$, i.e., $m_{\text{LoS}}^{(t)} \in \{0, 1, 2, \dots, m_{\text{LoS}}^{(t),\text{ref}}\}$.

Compared to the original action space $\mathcal{A}$ given by (5.8b) and (5.8c), the new action space $\mathcal{A}^{(t)}$ obtained above has a smaller size since $m_{\text{NLoS}}^{(t),\text{ref}} \geq 0$ and $m_{\text{LoS}}^{(t),\text{ref}} \leq M_{\text{NLoS}}$, which reduces the difficulty in solving the MDP problem.

5.4.2 DRL Algorithm

We adopt the deep Q -learning algorithm to solve the MDP problem. A deep neural network (DNN), characterized by parameters $\boldsymbol{\theta}$, is constructed to approximate the Q value of each state-action pair, denoted by $Q(\mathbf{s}^{(t)}, \mathbf{a}^{(t)} | \boldsymbol{\theta})$. At the beginning of the t -th time interval, with a probability $1 - \epsilon$, the action corresponding to the maximal approximated Q value is selected, i.e., $\mathbf{a}^{(t),*} = \operatorname{argmax}_{\mathbf{a}^{(t)} \in \mathcal{A}^{(t)}} Q(\mathbf{s}^{(t)}, \mathbf{a}^{(t)} | \boldsymbol{\theta})$, and otherwise, $\mathbf{a}^{(t),*}$ is randomly selected. After taking the action $\mathbf{a}^{(t),*}$ and observing the next state and the instantaneous reward, the approximation of the Q value, corresponding to the state-action pair $(\mathbf{s}^{(t)}, \mathbf{a}^{(t),*})$, is updated. The parameters in the DNN, i.e., $\boldsymbol{\theta}$, are adjusted to minimize the difference between the Q values before and after the update.

5.5 Simulation Results

We consider a $1000 \text{ m} \times 1000 \text{ m}$ area, where one AP is located in the center of the area, and 60 virtual content items are randomly positioned in the corresponding virtual space. In each time interval, the blockages of the signal propagation paths, i.e., $\xi_k^{(t)}$ for all $k \in \mathcal{K}^{(u(t))}$, follow independent and identical Bernoulli distributions, with the mean referred to as the path blockage probability. The initial location of an AR user and the order in which the virtual content items are viewed by the AR user are randomly determined. In viewing each virtual content item, the user trajectory is generated by randomly selecting from the public AR user trace dataset (https://github.com/Yong-Chen94/6DoF_Video_FoV_Dataset). We generate light intensities, ranging from 0 to 1, for all the locations within the area. In the t -th time interval, if the user is at a location where the light intensity is below a threshold, set as 0.4, the on-device user positioning is considered to malfunction, and the sensing-based user positioning is required, i.e., $\eta^{(t)} = 1$, and otherwise, we have $\eta^{(t)} = 0$. The DNN for Q value approximation consists of four fully-connected hidden layers with 400, 200, 100, and 20 neurons, respectively. The other main simulation settings are given in Table 5.1.

Table 5.1: Main simulation settings in sensing-assisted mobile AR streaming

Parameter	Value
Total Bandwidth, B	150 MHz
Total subcarrier number, $N_s^{\max}$	1×10^4
Sensing subcarrier number, N_s^{sen}	1×10^3
Number of OFDM symbols per slot, N_t	14
Total effective isotropic radiated power, $P_T G_T$	30 dBm
Receive antenna number, N_R	4
Number of VAs, $K^{(u)}$, $\forall u \in \mathcal{U}$	10
Power spectral density of noise, N_0	$4 \times 10^{-20} \text{ W/Hz}$
Vertical and Horizontal FoV, $\{\theta_{\text{FoV}_v}, \theta_{\text{FoV}_h}\}$	$\{45^\circ, 45^\circ\}$
Height of the AR device, h_u	1.7 m
Length and Height of AR contents, $\{h_c^{\max}, l_c^{\max}\}$	$\{2.0 \text{ m } 2.0 \text{ m}\}$

We compare our proposed scheme with the following two schemes: 1) LoS-Prone Greedy: the reference decision in (5.11), which minimizes the resource consumption in only the current time interval, is used. Due to the generally greater signal attenuation and the position uncertainty of VAs corresponding to NLoS paths, it is prone to transmit sensing signals only along the LoS path; 2) Vanilla DRL: a deep Q -learning algorithm with the same setting to our proposed scheme is used to solve the MDP problem with the original action space.

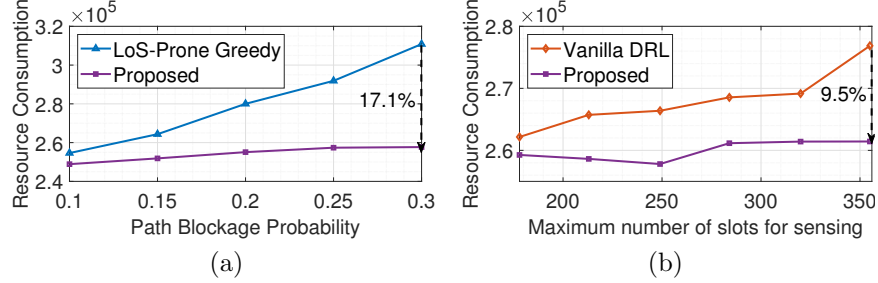


Fig. 5.4: Average resource consumption per time interval.

In Fig. 5.4(a), under varying path blockage probabilities, we show the average resource consumption in communication REs for service provision to the AR user per time interval. It can be observed that our proposed scheme results in lower resource consumption compared to the LoS-prone greedy scheme. In addition, the reduction in resource consumption increases with the path blockage probability. This is because the LoS-prone greedy scheme mainly leverages the LoS path while the proposed scheme is prone to update the environment map so that the multiple NLoS paths can be additionally used for accurate user positioning. With more accurate user positioning, the uncertainty of the visible AR content is reduced, and a smaller region of AR content needs to be delivered to the user for a satisfactory viewing experience. In Fig. 5.4(b), we show the average resource consumption under varying maximum numbers of slots for sensing along the LoS path, i.e., M_{LoS} . It can be observed that the proposed scheme results in lower resource consumption compared to the vanilla DRL scheme, and the reduction in the resource consumption increases with M_{LoS} in most cases. This is because the proposed scheme captures the long-term benefit of environment mapping for user positioning to effectively reduce the action space, and a larger value of M_{LoS} indicates a larger size of the original action space.

5.6 Conclusion

In this chapter, we have proposed a model-assisted learning scheme for resource allocation in environment-aware mobile AR content delivery. The proposed scheme adaptively strikes the balance between the benefit and overhead of environment mapping for resource-efficient content delivery to an individual AR user. This scheme introduces a novel approach that leverages both mathematical modeling and deep reinforcement learning for resource allocation in complex scenarios.

Chapter 6

Conclusions and Future Works

In this chapter, we provide a summary of the research contributions of this thesis and identify several future research directions.

6.1 Main Research Contributions

In this thesis, we have studied the resource allocation and task scheduling for ISAC in future 6G networks, exploring the co-design and mutual benefits of SAC functionalities for efficiently satisfying the SAC demands. Specifically, we have identified three research problems and developed efficient solutions. First, a proactive resource management scheme has been designed to efficiently coordinate mobile devices and network infrastructures in SAC service provision. Second, an approach to offloading DNN inference-based sensing tasks has been proposed to cope with highly dynamic task arrivals and server workload. Third, a resource allocation scheme has been designed for adaptive environment mapping and user positioning in user-position-dependent AR streaming. The main contributions of this thesis can be summarized as follows.

- 1) A proactive resource management scheme has been developed for the efficient coordination of mobile devices and network infrastructures in SAC service provision. In the proposed scheme, individual network slices have been constructed to provide SAC services, and DTs of these network slices have been established to predict the non-stationary spatial distributions of devices and sensing targets. Based on the DTs, we have developed a closed-form and reliable decision-making strategy for proactive

resource management, including sensing region partitioning and radio and computing resource reservation. The work empowers the efficient implementation of ISAC in large-scale and non-stationary wireless networks.

- 2) An adaptive approach to offloading DNN-based sensing data processing tasks to edge computing servers has been proposed. This approach has incorporated the multi-layer and multi-exit architecture of the considered DNNs to enable the adaptive partitioning and offloading of individual sensing tasks. We have established DTs to monitor and emulate the task processing status and on-device and edge workload for providing sufficient data to a learning-based task offloading algorithm. This work presents an effective solution for optimizing sensing task performances by accurately characterizing task dynamics and efficiently leveraging the computing resources in wireless networks.
- 3) A radio resource allocation scheme was designed for efficient sensing-assisted AR streaming. In this scheme, the benefits of environment mapping for user positioning and content visibility in non-line-of-sight conditions have been mathematically characterized. With the characterization, a reference decision-making strategy has been identified and used to guide a learning-based decision-making strategy for environment mapping, user positioning, and content delivery. The work provided valuable insights for efficiently acquiring information on users and the surrounding environment through sensing to optimize communication service provision.

In summary, this thesis has explored the co-designs and mutual benefits of SAC functionalities for efficient resource management, optimizing SAC task performance. To achieve this goal, the three research works have been built upon a slicing-based resource management framework, where proactive and real-time resource allocation and task scheduling can be adaptively determined. In addition, data-driven approaches for resource management in ISAC have been proposed, which are empowered by the collected and augmented data from DTs of network entities and the guidance from mathematical model-based resource allocation decisions. This thesis offers insights and practical guidelines for the efficient implementation of ISAC in 6G and beyond.

6.2 Future Research Directions

The integration of physical and virtual worlds is expected to be increasingly seamless and pervasive in the future, which poses stricter and more diverse requirements on SAC

functionalities. To this end, future research should focus on augmenting SAC functionalities and realizing personalized SAC service provision. To achieve the goals, we identify the potential of Generative Artificial Intelligence (GAI) and user Quality of Experience (QoE)-oriented resource management, respectively.

- GAI-empowered SAC:** In recent years, GAI has emerged as an evolutionary paradigm shift in AI techniques. Typically, GAI models concatenate a large number of transformers that excel in temporal sequence-to-sequence prediction and capture complex data distributions through training with a massive amount of data. As a result, based on user input, such as texts, audio, and images, GAI models can accurately predict the outcome that the user desires in an auto-regressive manner. The content generation capability of GAI can be beneficial for both SAC functionalities. In particular, in the sensing task of obtaining a map of the physical or propagation environment, typically only partial environment measurements from specific locations in the map can be obtained. As a result, generating a high-resolution and high-quality map from the noisy and partial observations is necessary yet challenging. To address this challenge, GAI, particularly diffusion models, has been studied and demonstrated its outstanding performance [151, 152, 153]. The performance of communication tasks can also be enhanced by the generation capability of GAI. Particularly, the semantic communication paradigm has been recently proposed, where only the intended meaning or information, rather than just the exact bits or symbols, is transmitted to reduce communication overhead. Nevertheless, the efficient extraction and reconstruction of semantic information is challenging [154]. To address this challenge, GAI trained on large-scale multi-modal corpora can be employed to extract shared semantic features across different modalities and facilitate robust reconstruction of the intended content [155]. Despite the potential, the implementation of GAI requires a substantial amount of computing and storage resources. Optimizing the GAI model design to facilitate efficient inference and training in wireless networks deserves further investigation. Moreover, balancing the computing overhead and the gain of GAI in enhancing SAC service provision requires an accurate characterization of the gain and efficient scheduling of GAI model inference.
- User QoE-oriented resource management for ISAC:** The SAC demands are expected to be increasingly pervasive and substantial with the proliferation and diversification of environment-aware mobile applications. Such demands may not be efficiently satisfied by the conventional Quality of Service (QoS)-oriented resource management schemes. This is because in the schemes, quality guarantees are provided for each service, such as those corresponding to data rate, delay, and packet

loss, rather than individual users' experience of the service. Characterizing users' QoE can help accurately identify the resources for the user to have a satisfactory experience, which further guides the efficient resource allocation to the users. Nevertheless, QoE-oriented resource management for ISAC is challenging from mainly three perspectives. First, the metrics that represent user QoE should be accurately identified. Those metrics could be performance metrics of the user application-level tasks that leverage the information gathered by SAC functionalities. However, identifying the tasks and measuring task performance can be challenging, since such information is usually not explicitly shared by users in wireless networks. Second, the relation between the QoE metrics and the resource allocation is difficult to characterize, since the dynamic user and device context in the specific application scenarios would also affect the QoE. Third, the models to estimate various QoE metrics can have complex and diverse mathematical expressions, which prevent efficient resource management. Addressing these challenges requires the synergy of the QoE metric identification, QoE modeling, and QoE-oriented resource management schemes.

References

- [1] S. He, K. Shi, C. Liu, B. Guo, J. Chen, and Z. Shi, “Collaborative sensing in Internet of Things: A comprehensive survey,” *IEEE Commun. Surveys Tuts.*, vol. 24, no. 3, pp. 1435–1474, 2022.
- [2] J. Katrasnik, F. Pernus, and B. Likar, “A survey of mobile robots for distribution power line inspection,” *IEEE Trans. Power Del.*, vol. 25, no. 1, pp. 485–493, 2010.
- [3] F. Liu, Y. Cui, C. Masouros, J. Xu, T. X. Han, Y. C. Eldar, and S. Buzzi, “Integrated sensing and communications: Toward dual-functional wireless networks for 6G and beyond,” *IEEE J. Sel. Areas Commun.*, vol. 40, no. 6, pp. 1728–1767, 2022.
- [4] 3rd Generation Partnership Project (3GPP), “Study on integrated sensing and communication (ISAC),” Tech. Rep. TR 22.837, 2024, release 19. Available on: <https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=4044>.
- [5] Z. Fan, L. Zhang, X. Wang, Y. Shen, and F. Deng, “LiDAR, IMU, and camera fusion for simultaneous localization and mapping: a systematic review,” *Artificial Intelligence Review*, vol. 58, no. 6, pp. 1–59, 2025.
- [6] X. Ye, “Cooperative sensing and computation for environment perception in autonomous driving with vehicular edge computing,” MASc Thesis, University of Waterloo, Waterloo, Canada, May 2022. [Online]. Available: <http://hdl.handle.net/10012/18239>
- [7] X. Luo, Q. Lin, R. Zhang, H.-H. Chen, X. Wang, and M. Huang, “ISAC—A survey on its layered architecture, technologies, standardizations, prototypes and testbeds,” *IEEE Commun. Surveys Tuts.*, pp. 1–41, 2025, to be published, doi: 10.1109/COMST.2025.3565534.

- [8] S. Lu, F. Liu, Y. Li, K. Zhang, H. Huang, J. Zou, X. Li, Y. Dong, F. Dong, J. Zhu, Y. Xiong, W. Yuan, Y. Cui, and L. Hanzo, “Integrated sensing and communications: Recent advances and ten open challenges,” *IEEE Internet Things J.*, vol. 11, no. 11, pp. 19 094–19 120, 2024.
- [9] C. Liu, W. Yuan, S. Li, X. Liu, H. Li, D. W. K. Ng, and Y. Li, “Learning-based predictive beamforming for integrated sensing and communication in vehicular networks,” *IEEE J. Sel. Areas Commun.*, vol. 40, no. 8, pp. 2317–2334, 2022.
- [10] B. Han, Y. Liu, and F. Qian, “Vivo: Visibility-aware mobile volumetric video streaming,” in *Proc. Annu. Int. Conf. Mob. Comput. Netw.*, 2020, pp. 1–13.
- [11] I. T. U. (ITU), “Framework and overall objectives of the future development of IMT for 2030 and beyond,” Tech. Rep. Recommendation ITU-R M.2160-0, Dec. 2023, available on: https://www.itu.int/dms_pubrec/itu-r/rec/m/R-REC-M.2160-0-202311-I%21%21PDF-E.pdf.
- [12] 3rd Generation Partnership Project (3GPP), “Moderated discussion for Rel-20 integrated sensing and communication,” Feb. 2025, version 0.0.2, TSG SA WG2 Meeting #167. Available on: https://www.3gpp.org/ftp/tsg_sa/wg2_arch/TSGS2_167_Athens_2025-02/INBOX/DRAFTS/Rel-20%20ISAC/Moderated_discussion_for_Rel-20_Integrated_Sensing_and_Communication-v0.0.2.pdf.
- [13] H. Technologies, “Integrated sensing and communication: Towards 6g and beyond,” 2022, available on: <https://www.huawei.com/en/huaweitech/future-technologies/6g-isac-thz>.
- [14] China Unicom and ZTE, “China Unicom Henan, Yunhuan Connected Drone Technology Co., Ltd. and ZTE complete 5G-A low-altitude ISAC commercial deployment for blood delivery services,” Nov. 2024, available on: <https://www.zte.com.cn/global/about/news/china-unicom-henan-yunhuan-connected-drone-technology-co-ltd-and-zte-complete-5g-a-low-altitude-isac-commercial-deployment-for-blood-delivery-services.html>.
- [15] H. Zhang, S. Mao, D. Niyato, and Z. Han, “Location-dependent augmented reality services in wireless edge-enabled metaverse systems,” *IEEE Open J. Commun. Soc.*, vol. 4, pp. 171–183, 2023.
- [16] J. Leng, W. Sha, B. Wang, P. Zheng, C. Zhuang, Q. Liu, T. Wuest, D. Mourtzis, and L. Wang, “Industry 5.0: Prospect and retrospect,” *J. Manuf. Syst.*, vol. 65, pp. 279–295, 2022.

- [17] J. M. Rožanec, I. Novalija, P. Zajec, K. Kenda, H. Tavakoli Ghinani, S. Suh, E. Veliou, D. Papamartzivanos, T. Giannetsos, S. A. Menesidou *et al.*, “Human-centric artificial intelligence architecture for industry 5.0 applications,” *Int. J. Prod. Res.*, vol. 61, no. 20, pp. 6847–6872, 2023.
- [18] X. Shen, J. Gao, M. Li, C. Zhou, S. Hu, M. He, and W. Zhuang, “Toward immersive communications in 6G,” *Front. Comput. Sci.*, vol. 4, 2023.
- [19] J. Li, C. Zhang, Z. Liu, R. Hong, and H. Hu, “Optimal volumetric video streaming with hybrid saliency based tiling,” *IEEE Trans. Multimedia*, vol. 25, pp. 2939–2953, 2022.
- [20] H. Chen, B. Xu, S. Wang, X. Cao, and Z. Ma, “Toward adaptive volumetric video streaming: A joint network-viewport adaptation framework,” *IEEE Commun. Mag.*, vol. 63, no. 3, pp. 197–203, 2025.
- [21] N. Kan, J. Zou, C. Li, W. Dai, and H. Xiong, “Rapt360: Reinforcement learning-based rate adaptation for 360-degree video streaming with adaptive prediction and tiling,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 3, pp. 1607–1623, 2022.
- [22] Q. Ye, W. Zhuang, S. Zhang, A.-L. Jin, X. Shen, and X. Li, “Dynamic radio resource slicing for a two-tier heterogeneous wireless network,” *IEEE Trans. Veh. Technol.*, vol. 67, no. 10, pp. 9896–9910, 2018.
- [23] J. Chen, X. Wang, and Y.-C. Liang, “Impact of channel aging on dual-function radar-communication systems: Performance analysis and resource allocation,” *IEEE Trans. Commun.*, vol. 71, no. 8, pp. 4972–4987, 2023.
- [24] W. Yi, Y. Yuan, R. Hoseinnezhad, and L. Kong, “Resource scheduling for distributed multi-target tracking in netted colocated MIMO radar systems,” *IEEE Trans. Signal Process.*, vol. 68, pp. 1602–1617, 2020.
- [25] N. C. Luong, X. Lu, D. T. Hoang, D. Niyato, and D. I. Kim, “Radio resource management in joint radar and communication: A comprehensive survey,” *IEEE Commun. Surveys Tuts.*, vol. 23, no. 2, pp. 780–814, 2021.
- [26] Z. Xie, R. Li, Z. Jiang, J. Zhu, X. She, and P. Chen, “Optimal scheduling policy for time-division joint radar and communication systems: Cross-layer design and sensing for free,” *IEEE Internet Things J.*, vol. 10, no. 23, pp. 20 746–20 760, 2023.

- [27] H. Ju, Y. Long, X. Fang, Y. Fang, and R. He, “Adaptive scheduling for joint communication and radar detection: Tradeoff among throughput, delay, and detection performance,” *IEEE Trans. Veh. Technol.*, vol. 71, no. 1, pp. 670–680, 2022.
- [28] F. Dong, F. Liu, Y. Cui, W. Wang, K. Han, and Z. Wang, “Sensing as a service in 6G perceptive networks: A unified framework for ISAC resource allocation,” *IEEE Trans. Wireless Commun.*, vol. 22, no. 5, pp. 3522–3536, 2023.
- [29] K. Meng, C. Masouros, G. Chen, and F. Liu, “Network-level integrated sensing and communication: Interference management and BS coordination using stochastic geometry,” *IEEE Trans. Wireless Commun.*, vol. 23, no. 12, pp. 19 365–19 381, 2024.
- [30] R. Li, Q. Zhang, D. Ma, K. Yu, and Y. Huang, “Joint target assignment and resource allocation for multi-base station cooperative ISAC in AAV detection,” *IEEE Trans. Veh. Technol.*, vol. 74, no. 5, pp. 7700–7714, 2025.
- [31] L. Zhao, D. Wu, L. Zhou, and Y. Qian, “Radio resource allocation for integrated sensing, communication, and computation networks,” *IEEE Trans. Wireless Commun.*, vol. 21, no. 10, pp. 8675–8687, 2022.
- [32] Y. Kang, J. Hauswald, C. Gao, A. Rovinski, T. Mudge, J. Mars, and L. Tang, “Neurosurgeon: Collaborative intelligence between the cloud and mobile edge,” *ACM SIGARCH Comput. Archit. News*, vol. 45, no. 1, pp. 615–629, Mar. 2017.
- [33] J. Yan, S. Bi, and Y.-J. A. Zhang, “Optimal model placement and online model splitting for device-edge co-inference,” *IEEE Trans. Wireless Commun.*, vol. 21, no. 10, pp. 8354–8367, Oct. 2022.
- [34] M. Gao, R. Shen, L. Shi, W. Qi, J. Li, and Y. Li, “Task partitioning and offloading in DNN-task enabled mobile edge computing networks,” *IEEE Trans. Mobile Comput.*, vol. 22, no. 4, pp. 2435–2445, Apr. 2023.
- [35] Y. He, G. Yu, Y. Cai, and H. Luo, “Integrated sensing, computation, and communication: System framework and performance optimization,” *IEEE Trans. Wireless Commun.*, vol. 23, no. 2, pp. 1114–1128, 2024.
- [36] Y. Zeng, J. Chen, J. Xu, D. Wu, X. Xu, S. Jin, X. Gao, D. Gesbert, S. Cui, and R. Zhang, “A tutorial on environment-aware communications via channel knowledge map for 6G,” *IEEE Commun. Surveys Tuts.*, vol. 26, no. 3, pp. 1478–1519, 2024.

- [37] X. Shen, J. Gao, W. Wu, K. Lyu, M. Li, W. Zhuang, X. Li, and J. Rao, "AI-assisted network-slicing based next-generation wireless networks," *IEEE Open J. Veh. Technol.*, vol. 1, pp. 45–66, 2020.
- [38] X. Shen, J. Gao, W. Wu, M. Li, C. Zhou, and W. Zhuang, "Holistic network virtualization and pervasive network intelligence for 6G," *IEEE Commun. Surveys Tuts.*, vol. 24, no. 1, pp. 1–30, Firstquarter 2022.
- [39] A. Hassanien, M. G. Amin, Y. D. Zhang, and F. Ahmad, "Signaling strategies for dual-function radar communications: an overview," *IEEE Aerosp. Electron. Syst. Mag.*, vol. 31, no. 10, pp. 36–45, 2016.
- [40] M. Li, J. Gao, C. Zhou, L. Zhao, and X. Shen, "Digital-twin-empowered resource allocation for on-demand collaborative sensing," *IEEE Internet Things J.*, vol. 11, no. 23, pp. 37 942–37 958, 2024.
- [41] M. L. Rahman, J. A. Zhang, X. Huang, Y. J. Guo, and R. W. Heath, "Framework for a perceptive mobile network using joint communication and radar sensing," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 56, no. 3, pp. 1926–1941, 2020.
- [42] Z. Guang, Y. Liu, Q. Wu, W. Wang, and Q. Shi, "Power minimization for isac system using beyond diagonal reconfigurable intelligent surface," *IEEE Trans. Veh. Technol.*, vol. 73, no. 9, pp. 13 950–13 955, 2024.
- [43] Q. Zhang, W. Xiao, P. Zhu, Y. Yao, W. Zhang, and F. Shu, "Transmit power minimization for double-RIS-enabled multi-user ISAC system in vehicular networks," *IEEE Trans. Intell. Transp. Syst.*, pp. 1–14, 2025.
- [44] Z. He, W. Xu, H. Shen, D. W. K. Ng, Y. C. Eldar, and X. You, "Full-duplex communication for ISAC: Joint beamforming and power optimization," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 9, pp. 2920–2936, 2023.
- [45] J. Chen, X. Wang, and D. Niyato, "Beyond ISAC: Toward integrated heterogeneous service provisioning via elastic multi-dimensional multiple access," 2025. [Online]. Available: <https://arxiv.org/abs/2504.11692>
- [46] J. Chen and X. Wang, "Radiation footprint control in cell-free cooperative isac: Optimal joint bs activation and beamforming coordination," 2025. [Online]. Available: <https://arxiv.org/abs/2504.10830>

- [47] B. Li and X. Wang, “Opportunistic reuse of spatial-temporal resources in multi-user isac systems for value of service maximization,” *IEEE Trans. Wireless Commun.*, pp. 1–15, 2025, to be published, doi: 10.1109/TWC.2025.3573209.
- [48] P. Liu, G. Zhu, S. Wang, W. Jiang, W. Luo, H. V. Poor, and S. Cui, “Toward ambient intelligence: Federated edge learning with task-oriented sensing, computation, and communication integration,” *IEEE J. Sel. Topics Signal Process.*, vol. 17, no. 1, pp. 158–172, 2023.
- [49] S. M. Kay, *Fundamentals of statistical signal processing: estimation theory*. Prentice-Hall, Inc., 1993.
- [50] K. Qu and W. Zhuang, “Scalable and dynamic cooperative perception: A data/model co-driven framework,” *IEEE Netw.*, vol. 38, no. 6, pp. 178–186, 2024.
- [51] X. Ye, K. Qu, W. Zhuang, and X. Shen, “Accuracy-aware cooperative sensing and computing for connected autonomous vehicles,” *IEEE Trans. Mobile Comput.*, vol. 23, no. 8, pp. 8193–8207, 2024.
- [52] Q. Zhang, H. Sun, X. Gao, X. Wang, and Z. Feng, “Time-division ISAC enabled connected automated vehicles cooperation algorithm design and performance evaluation,” *IEEE J. Sel. Areas Commun.*, vol. 40, no. 7, pp. 2206–2218, 2022.
- [53] Z. He, W. Xu, H. Shen, D. W. K. Ng, Y. C. Eldar, and X. You, “Full-duplex communication for ISAC: Joint beamforming and power optimization,” *IEEE J. Sel. Areas Commun.*, vol. 41, no. 9, pp. 2920–2936, 2023.
- [54] Y. Hmamouche, M. Benjillali, S. Saoudi, H. Yanikomeroglu, and M. D. Renzo, “New trends in stochastic geometry for wireless networks: A tutorial and survey,” *Proc. IEEE*, vol. 109, no. 7, pp. 1200–1252, 2021.
- [55] A. Munari, L. Simić, and M. Petrova, “Stochastic geometry interference analysis of radar network performance,” *IEEE Commun. Lett.*, vol. 22, no. 11, pp. 2362–2365, 2018.
- [56] M. Mei, M. Yao, Q. Yang, J. Wang, and R. R. Rao, “Stochastic network calculus analysis of spatial-temporal integrated sensing and communication networks,” *IEEE Trans. Veh. Technol.*, vol. 73, no. 6, pp. 9120–9124, 2024.
- [57] H. Zhang, Y. Zhang, X. Liu, C. Ren, H. Li, and C. Sun, “Time allocation approaches for a perceptive mobile network using integration of sensing and communication,” *IEEE Trans. Wireless Commun.*, vol. 23, no. 2, pp. 1158–1169, 2024.

- [58] T. H. Luan, X. Ling, and X. Shen, “MAC in motion: Impact of mobility on the MAC of drive-thru Internet,” *IEEE Trans. Mobile Comput.*, vol. 11, no. 2, pp. 305–319, 2012.
- [59] J. Gao, W. Zhuang, M. Li, X. Shen, and X. Li, “MAC for machine-type communications in Industrial IoT—Part I: Protocol design and analysis,” *IEEE Internet Things J.*, vol. 8, no. 12, pp. 9945–9957, 2021.
- [60] J. Gao, M. Li, W. Zhuang, X. Shen, and X. Li, “MAC for machine-type communications in Industrial IoT—Part II: Scheduling and numerical results,” *IEEE Internet Things J.*, vol. 8, no. 12, pp. 9958–9969, 2021.
- [61] M. Li, J. Gao, X. Shen, and L. Zhao, *Intelligent computing and communication for the internet of vehicles*. Springer, 2023.
- [62] J. Chen and X. Wang, “Learning-based intermittent CSI estimation with adaptive intervals in integrated sensing and communication systems,” *IEEE J. Sel. Topics Signal Process.*, vol. 18, no. 5, pp. 917–932, 2024.
- [63] G. Li, S. Wang, J. Li, R. Wang, F. Liu, X. Peng, T. X. Han, and C. Xu, “Integrated sensing and communication from learning perspective: An SDP3 approach,” *IEEE Internet Things J.*, vol. 11, no. 4, pp. 5589–5603, 2024.
- [64] Y. Xu, D. Xu, and S. Song, “Sensing-assisted robust SWIPT for mobile energy harvesting receivers in networked ISAC systems,” *IEEE Trans. Wireless Commun.*, vol. 24, no. 3, pp. 2094–2109, 2025.
- [65] Y. Cui, H. Ding, L. Zhao, and J. An, “Integrated sensing and communication: A network level perspective,” *IEEE Wireless Commun.*, vol. 31, no. 1, pp. 103–109, 2024.
- [66] X. Wang, Z. Han, R. Xi, G. Liu, L. Han, J. Jin, Y. Xue, L. Ma, Y. Wang, T. Jiang, M. Lou, Q. Wang, and J. Wang, “Cooperative integrated sensing and communication in 6G: From operators perspective,” *IEEE Wireless Commun.*, vol. 32, no. 1, pp. 52–59, 2025.
- [67] Z. Wei, W. Jiang, Z. Feng, H. Wu, N. Zhang, K. Han, R. Xu, and P. Zhang, “Integrated sensing and communication enabled multiple base stations cooperative sensing towards 6g,” *IEEE Netw.*, vol. 38, no. 4, pp. 207–215, 2024.

- [68] K. Qu, W. Zhuang, Q. Ye, W. Wu, and X. Shen, “Model-assisted learning for adaptive cooperative perception of connected autonomous vehicles,” *IEEE Trans. Wireless Commun.*, vol. 23, no. 8, pp. 8820–8835, 2024.
- [69] A. Salem, K. Meng, C. Masouros, F. Liu, and D. Lopez-Perez, “Rethinking dense cells for integrated sensing and communications: A stochastic geometric view,” *IEEE Open J. Commun. Soc.*, vol. 5, pp. 2226–2239, 2024.
- [70] Z. Sun, S. Yan, N. Jiang, J. Zhou, and M. Peng, “Performance analysis of integrated sensing and communication networks with blockage effects,” *IEEE Trans. Veh. Technol.*, vol. 73, no. 11, pp. 16 876–16 891, 2024.
- [71] M. Li, J. Gao, L. Zhao, and X. Shen, “Deep reinforcement learning for collaborative edge computing in vehicular networks,” *IEEE Trans. Cogn. Commun. Netw.*, vol. 6, no. 4, pp. 1122–1135, Dec. 2020.
- [72] C. Zhou, W. Wu, H. He, P. Yang, F. Lyu, N. Cheng, and X. Shen, “Deep reinforcement learning for delay-oriented IoT task scheduling in SAGIN,” *IEEE Trans. Wireless Commun.*, vol. 20, no. 2, pp. 911–925, Feb. 2021.
- [73] C. Zhou, J. Gao, M. Li, X. Shen, and W. Zhuang, “Digital twin-empowered network planning for multi-tier computing,” *J. Commun. Inf. Netw.*, vol. 7, no. 3, pp. 221–238, 2022.
- [74] F. Lyu, J. Ren, P. Yang, N. Cheng, Y. Zhang, and X. Shen, “SoSA: Socializing static APs for edge resource pooling in large-scale wifi system,” in *Proc. IEEE Int. Conf. Comput. Commun. (INFOCOM)*, 2020, pp. 1181–1190.
- [75] M. He, H. Wu, X. Shen, and W. Zhuang, “Cooperative resource scheduling for environment sensing in satellite–terrestrial vehicular networks,” *IEEE Internet Things J.*, vol. 12, no. 6, pp. 6734–6748, 2025.
- [76] H. Wu, J. Chen, W. Xu, N. Cheng, W. Shi, L. Wang, and X. Shen, “Delay-minimized edge caching in heterogeneous vehicular networks: A matching-based approach,” *IEEE Trans. Wireless Commun.*, vol. 19, no. 10, pp. 6409–6424, 2020.
- [77] M. He, C. Zhou, H. Wu, and X. Sherman Shen, “Learning-based cache placement and content delivery for satellite-terrestrial integrated networks,” in *Proc. IEEE Glob. Commun. Conf. (GLOBECOM)*, 2021, pp. 1–6.

- [78] Y. Pei, M. Li, H. Wu, Q. Ye, C. Zhou, S. Hu, and X. Shen, “Joint caching and computing resource reservation for edge-assisted location-aware augmented reality,” in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2023, pp. 2547–2552.
- [79] F. Wu, F. Lyu, J. Ren, P. Yang, K. Qian, S. Gao, and Y. Zhang, “Characterizing internet card user portraits for efficient churn prediction model design,” *IEEE Trans. Mobile Comput.*, vol. 23, no. 2, pp. 1735–1752, 2023.
- [80] S. Duan, F. Lyu, J. Zhang, H. Lu, P. Yang, H. Wu, Y. Zhang, and X. Shen, “Moco: Urban user mobile contact detection based on cellular signaling trace,” *IEEE Trans. Mobile Comput.*, vol. 24, no. 8, pp. 6780–6796, 2025.
- [81] S. Duan, F. Lyu, X. Zhu, Y. Ding, H. Wang, D. Zhang, X. Liu, Y. Zhang, and J. Ren, “Velp: vehicle loading plan learning from human behavior in nationwide logistics system,” *Proc. VLDB Endow.*, vol. 17, no. 2, pp. 241–249, 2023.
- [82] Z. Lin, S. Bi, and Y.-J. A. Zhang, “Optimizing AI service placement and resource allocation in mobile edge intelligence systems,” *IEEE Trans. Wireless Commun.*, vol. 20, no. 11, pp. 7257–7271, Nov. 2021.
- [83] B. Yang, X. Cao, X. Li, Q. Zhang, and L. Qian, “Mobile-edge-computing-based hierarchical machine learning tasks distribution for IIoT,” *IEEE Internet Things J.*, vol. 7, no. 3, pp. 2169–2180, Mar. 2020.
- [84] W. Fan, Z. Chen, Y. Su, F. Wu, B. Tang, and Y. Liu, “Accuracy-based task offloading and resource allocation for edge intelligence in IoT,” *IEEE Wireless Commun. Lett.*, vol. 11, no. 2, pp. 371–375, Feb. 2022.
- [85] W. Wu, P. Yang, W. Zhang, C. Zhou, and X. Shen, “Accuracy-guaranteed collaborative DNN inference in industrial IoT via deep reinforcement learning,” *IEEE Trans. Ind. Informat.*, vol. 17, no. 7, pp. 4988–4998, July 2020.
- [86] W. Zhang, D. Yang, H. Peng, W. Wu, W. Quan, H. Zhang, and X. Shen, “Deep reinforcement learning based resource management for DNN inference in industrial IoT,” *IEEE Trans. Veh. Technol.*, vol. 70, no. 8, pp. 7605–7618, Aug. 2021.
- [87] S. Guo, C. Zhao, G. Wang, J. Yang, and S. Yang, “Ec²detect: Real-time online video object detection in edge-cloud collaborative iot,” *IEEE Internet Things J.*, vol. 9, no. 20, pp. 20 382–20 392, Oct. 2022.

- [88] K. Qu, W. Zhuang, W. Wu, M. Li, X. Shen, X. Li, and W. Shi, “Stochastic cumulative DNN inference with RL-aided adaptive IoT device-edge collaboration,” *IEEE Internet of Things J.*, pp. 1–1, 2023.
- [89] X. Tang, X. Chen, L. Zeng, S. Yu, and L. Chen, “Joint multiuser DNN partitioning and computational resource allocation for collaborative edge intelligence,” *IEEE Internet Things J.*, vol. 8, no. 12, pp. 9511–9522, June 2021.
- [90] H. Liang, Q. Sang, C. Hu, D. Cheng, X. Zhou, D. Wang, W. Bao, and Y. Wang, “DNN surgery: Accelerating DNN inference on the edge through layer partitioning,” *IEEE Trans. Cloud Comput.*, vol. 11, no. 3, pp. 3111–3125, 2023.
- [91] J. Song, Z. Liu, X. Wang, C. Qiu, and X. Chen, “Adaptive and collaborative edge inference in task stream with latency constraint,” in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2021, pp. 1–6.
- [92] W. Fan, Z. Chen, Z. Hao, Y. Su, F. Wu, B. Tang, and Y. Liu, “DNN deployment, task offloading, and resource allocation for joint task inference in IIoT,” *IEEE Trans. Ind. Informat.*, vol. 19, no. 2, pp. 1634–1646, 2023.
- [93] Y. Deng, F. Lyu, J. Ren, Y.-C. Chen, P. Yang, Y. Zhou, and Y. Zhang, “Fair: Quality-aware federated learning with precise user incentive and model aggregation,” in *Proc. IEEE Int. Conf. Comput. Commun. (INFOCOM)*, 2021, pp. 1–10.
- [94] Y. Deng, F. Lyu, T. Xia, Y. Zhou, Y. Zhang, J. Ren, and Y. Yang, “A communication-efficient hierarchical federated learning framework via shaping data distribution at edge,” *IEEE/ACM Trans. Netw.*, vol. 32, no. 3, pp. 2600–2615, 2024.
- [95] H. Lu, F. Lyu, H. Wu, J. Zhang, J. Ren, Y. Zhang, and X. Shen, “FL-AMM: Federated learning augmented map matching with heterogeneous cellular moving trajectories,” *IEEE J. Sel. Areas Commun.*, vol. 41, no. 12, pp. 3878–3892, 2023.
- [96] Y. Deng, F. Lyu, J. Ren, Y.-C. Chen, P. Yang, Y. Zhou, and Y. Zhang, “Improving federated learning with quality-aware user incentive and auto-weighted model aggregation,” *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 12, pp. 4515–4529, 2022.
- [97] W. Wu, M. Li, K. Qu, C. Zhou, X. Shen, W. Zhuang, X. Li, and W. Shi, “Split learning over wireless networks: Parallel design and resource management,” *IEEE J. Sel. Areas Commun.*, vol. 41, no. 4, pp. 1051–1066, 2023.

- [98] S. Zhang, G. Cheng, Z. Li, and W. Wu, “SplitLLM: Hierarchical split learning for large language model over wireless network,” 2025. [Online]. Available: <https://arxiv.org/abs/2501.13318>
- [99] Z. Li, S. Wu, L. Li, and S. Zhang, “Energy-efficient split learning for fine-tuning large language models in edge networks,” *IEEE Netw. Lett.*, pp. 1–5, 2025, to be published, doi: 10.1109/LNET.2025.3530430.
- [100] R. T. Mullapudi, S. Chen, K. Zhang, D. Ramanan, and K. Fatahalian, “Online model distillation for efficient video inference,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 3572–3581.
- [101] M. Khani, P. Hamadani, A. Nasr-Esfahany, and M. Alizadeh, “Real-time video inference on edge devices via adaptive model streaming,” in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2021, pp. 4552–4562.
- [102] S. Hu, J. Gao, X. Huang, M. Li, K. Qu, C. Zhou, and X. Shen, “Digital twin-based user-centric edge continual learning in integrated sensing and communication,” *arXiv preprint arXiv:2311.12223*, 2023.
- [103] K. Qu, W. Zhuang, Q. Ye, X. Shen, X. Li, and J. Rao, “Dynamic flow migration for embedded services in SDN/NFV-enabled 5G core networks,” *IEEE Trans. Commun.*, vol. 68, no. 4, pp. 2394–2408, 2020.
- [104] K. Qu, W. Zhuang, X. Shen, X. Li, and J. Rao, “Dynamic resource scaling for VNF over nonstationary traffic: A learning approach,” *IEEE Trans. Cogn. Commun. Netw.*, vol. 7, no. 2, pp. 648–662, 2021.
- [105] Q. Ye, W. Shi, K. Qu, H. He, W. Zhuang, and X. Shen, “Joint RAN slicing and computation offloading for autonomous vehicular networks: A learning-assisted hierarchical approach,” *IEEE Open J. Veh. Technol.*, vol. 2, pp. 272–288, 2021.
- [106] W. Wu, N. Chen, C. Zhou, M. Li, X. Shen, W. Zhuang, and X. Li, “Dynamic RAN slicing for service-oriented vehicular networks via constrained learning,” *IEEE J. Sel. Areas Commun.*, vol. 39, no. 7, pp. 2076–2089, 2021.
- [107] C. Zhou, J. Gao, M. Li, X. Shen, W. Zhuang, X. Li, and W. Shi, “AI-assisted slicing-based resource management for two-tier radio access networks,” *IEEE Trans. Cogn. Commun. Netw.*, vol. 9, no. 6, pp. 1691–1706, 2023.

- [108] M. Li, J. Gao, C. Zhou, X. Shen, and W. Zhuang, "Slicing-based artificial intelligence service provisioning on the network edge: Balancing AI service performance and resource consumption of data management," *IEEE Veh. Technol. Mag.*, vol. 16, no. 4, pp. 16–26, 2021.
- [109] M. A. Hossain, A. Xiang, A. Kiani, T. Saboorian, J. Kaippallimalil, and N. Ansari, "Ai-assisted e2e network slicing for integrated sensing and communication in 6g networks," *IEEE Internet of Things Journal*, vol. 11, no. 6, pp. 10 627–10 634, 2024.
- [110] C. Zhou, S. Hu, J. Gao, X. Huang, W. Zhuang, and X. Shen, "User-centric immersive communications in 6G: A data-oriented framework via digital twin," *IEEE Wireless Commun.*, vol. 32, no. 3, pp. 122–129, 2025.
- [111] M. Grieves and J. Vickers, "Digital twin: Mitigating unpredictable, undesirable emergent behavior in complex systems," in *Transdisciplinary perspectives on complex systems: New findings and approaches*. Cham: Springer, 2017, pp. 85–113.
- [112] S. Hu, M. Li, J. Gao, C. Zhou, and X. Shen, "Adaptive device-edge collaboration on DNN inference in AIoT: A digital twin-assisted approach," *IEEE Internet Things J.*, vol. 11, no. 7, pp. 12 893–12 908, 2024.
- [113] C. Zhou, J. Gao, M. Li, N. Cheng, X. Shen, and W. Zhuang, "Digital-twin-based 3-D map management for edge-assisted device pose tracking in mobile AR," *IEEE Internet Things J.*, vol. 11, no. 10, pp. 17 812–17 826, 2024.
- [114] P. Almasan, M. Ferriol-Galmés, J. Paillisse, J. Suárez-Varela, D. Perino, D. López, A. A. P. Perales, P. Harvey, L. Ciavaglia, L. Wong, V. Ram, S. Xiao, X. Shi, X. Cheng, A. Cabellos-Aparicio, and P. Barlet-Ros, "Network digital twin: Context, enabling technologies, and opportunities," *IEEE Commun. Mag.*, vol. 60, no. 11, pp. 22–27, 2022.
- [115] M. Abdel-Basset, H. Hawash, K. M. Sallam, I. Elgendi, and K. Munasinghe, "Digital twin for optimization of slicing-enabled communication networks: A federated graph learning approach," *IEEE Commun. Mag.*, vol. 61, no. 10, pp. 100–106, 2023.
- [116] H. Wang, Y. Wu, G. Min, and W. Miao, "A graph neural network-based digital twin for network slicing management," *IEEE Trans. Ind. Informat.*, vol. 18, no. 2, pp. 1367–1376, Feb. 2022.

- [117] M. He, H. Wu, C. Zhou, S. Hu, Z. Tang, and W. Zhuang, “Digital twin-assisted robust and adaptive resource slicing in LEO satellite networks,” in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, 2024, pp. 3261–3266.
- [118] K. Meng, C. Masouros, A. P. Petropulu, and L. Hanzo, “Cooperative ISAC networks: Opportunities and challenges,” *IEEE Wireless Commun.*, pp. 1–8, 2024, to be published, doi: 10.1109/MWC.008.2400151.
- [119] J. Li, X. Shao, F. Chen, S. Wan, C. Liu, Z. Wei, and D. Wing Kwan Ng, “Networked integrated sensing and communications for 6G wireless systems,” *IEEE Internet Things J.*, vol. 11, no. 17, pp. 29 062–29 075, 2024.
- [120] D. M. Manias, A. Chouman, and A. Shami, “Model drift in dynamic networks,” *IEEE Commun. Mag.*, vol. 61, no. 10, pp. 78–84, 2023.
- [121] X. Wang, L. Nkenyereye, S. Rani, and J. Lyu, “Adaptive sensing for Internet of robotic things platforms with integrated sensing, computing and communication capabilities,” *IEEE Internet Things J.*, 2024.
- [122] M. Goyal, A. Kumar, and V. Sharma, “Optimal cross-layer scheduling of transmissions over a fading multiaccess channel,” *IEEE Trans. Inf. Theory*, vol. 54, no. 8, pp. 3518–3537, 2008.
- [123] Y. Wang, Q. Zhang, Z. Wei, L. Kui, F. Liu, and Z. Feng, “Performance analysis of uncoordinated interference mitigation for automotive radar,” *IEEE Trans. Veh. Technol.*, vol. 72, no. 4, pp. 4222–4235, 2023.
- [124] G. Ghatak, R. Koirala, A. De Domenico, B. Denis, D. Dardari, B. Uguen, and M. Coupechoux, “Beamwidth optimization and resource partitioning scheme for localization assisted mm-wave communication,” *IEEE Trans. Commun.*, vol. 69, no. 2, pp. 1358–1374, 2021.
- [125] J. A. Marvel and R. Norcross, “Implementing speed and separation monitoring in collaborative robot workcells,” *Robot. Comput. Integr. Manuf.*, vol. 44, pp. 144–155, 2017.
- [126] J. A. González, F. J. Rodríguez-Cortés, O. Cronie, and J. Mateu, “Spatio-temporal point process statistics: a review,” *Spatial Statist.*, vol. 18, pp. 505–544, 2016.
- [127] P. D. Mankar, G. Das, and S. S. Pathak, “Modeling and coverage analysis of BS-centric clustered users in a random wireless network,” *IEEE Wireless Commun. Lett.*, vol. 5, no. 2, pp. 208–211, 2016.

- [128] X. Lu, M. Salehi, M. Haenggi, E. Hossain, and H. Jiang, “Stochastic geometry analysis of spatial-temporal performance in wireless networks: A tutorial,” *IEEE Commun. Surveys Tuts.*, vol. 23, no. 4, pp. 2753–2801, 2021.
- [129] J. Zhang and D. Tao, “Empowering things with intelligence: A survey of the progress, challenges, and opportunities in artificial intelligence of things,” *IEEE Internet Things J.*, vol. 8, no. 10, pp. 7789–7817, May 2021.
- [130] X. Chen, J. Zhang, B. Lin, Z. Chen, K. Wolter, and G. Min, “Energy-efficient offloading for DNN-based smart IoT systems in cloud-edge environments,” *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 3, pp. 683–697, Mar. 2021.
- [131] Y. Huang, X. Qiao, J. Tang, P. Ren, L. Liu, C. Pu, and J. Chen, “An integrated cloud-edge-device adaptive deep learning service for cross-platform web,” *IEEE Trans. Mobile Comput.*, vol. 22, no. 4, pp. 1950–1967, Oct. 2023.
- [132] L. Zeng, E. Li, Z. Zhou, and X. Chen, “Boomerang: On-demand cooperative deep neural network inference for edge intelligence on the industrial Internet of Things,” *IEEE Netw.*, vol. 33, no. 5, pp. 96–103, Sept.-Oct. 2019.
- [133] B. Han, V. Sciancalepore, Y. Xu, D. Feng, and H. D. Schotten, “Impatient queuing for intelligent task offloading in multiaccess edge computing,” *IEEE Trans. Wireless Commun.*, vol. 22, no. 1, pp. 59–72, Jan. 2023.
- [134] L. U. Khan, W. Saad, D. Niyato, Z. Han, and C. S. Hong, “Digital-twin-enabled 6G: Vision, architectural trends, and future directions,” *IEEE Commun. Mag.*, vol. 60, no. 1, pp. 74–80, Jan. 2022.
- [135] Y. Lu, S. Maharjan, and Y. Zhang, “Adaptive edge association for wireless digital twin networks in 6G,” *IEEE Internet Things J.*, vol. 8, no. 22, pp. 16 219–16 230, Nov. 2021.
- [136] W. Sun, H. Zhang, R. Wang, and Y. Zhang, “Reducing offloading latency for digital twin edge networks in 6G,” *IEEE Trans. Veh. Technol.*, vol. 69, no. 10, pp. 12 240–12 251, Oct. 2020.
- [137] W. Shi, S. Zhou, Z. Niu, M. Jiang, and L. Geng, “Multiuser co-inference with batch processing capable edge server,” *IEEE Trans. Wireless Commun.*, vol. 22, no. 1, pp. 286–300, 2023.

- [138] S. Teerapittayanon, B. McDanel, and H.-T. Kung, “Branchynet: Fast inference via early exiting from deep neural networks,” in *Proc. 23rd Int. Conf. Pattern Recognit. (ICPR)*, 2016, pp. 2464–2469.
- [139] X. Deng, J. Li, C. Ma, K. Wei, L. Shi, M. Ding, and W. Chen, “Low-latency federated learning with DNN partition in distributed industrial IoT networks,” *IEEE J. Sel. Areas Commun.*, vol. 41, no. 3, pp. 755–775, Dec. 2023.
- [140] C. Herrera, F. Krach, P. Ruyssen, and J. Teichmann, “Optimal stopping via randomized neural networks,” *arXiv:2104.13669*, 2021, Available: <https://arxiv.org/abs/2104.13669>.
- [141] T. S. Ferguson, “Optimal stopping and applications,” Available: <https://www.math.ucla.edu/~tom/Stopping/Contents.html>.
- [142] S. Becker, P. Cheridito, and A. Jentzen, “Deep optimal stopping,” *J. Mach. Learn. Res.*, vol. 20, pp. 1–25, Apr. 2019.
- [143] J. Tsitsiklis and B. Van Roy, “Regression methods for pricing complex american-style options,” *IEEE Trans. Neural Netw.*, vol. 12, no. 4, pp. 694–703, July 2001.
- [144] Z. Wu, J. Wen, Y. Xu, J. Yang, X. Li, and D. Zhang, “Enhanced spatial feature learning for weakly supervised object detection,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 1, pp. 961–972, 2024.
- [145] Y. Pei, M. Li, X. Huang, and X. Shen, “QoE-aware volumetric video caching and rendering for mobile extended reality services,” *IEEE Internet Things J.*, pp. 1–14, 2025.
- [146] Z. Liu, Q. Li, X. Chen, C. Wu, S. Ishihara, J. Li, and Y. Ji, “Point cloud video streaming: Challenges and solutions,” *IEEE Netw.*, vol. 35, no. 5, pp. 202–209, 2021.
- [147] Y. Zeng, J. Chen, J. Xu, D. Wu, X. Xu, S. Jin, X. Gao, D. Gesbert, S. Cui, and R. Zhang, “A tutorial on environment-aware communications via channel knowledge map for 6G,” *IEEE Commun. Surv. Tutor.*, vol. 26, no. 3, pp. 1478–1519, 2024.
- [148] S. Aditya, A. F. Molisch, and H. M. Behairy, “A survey on the impact of multipath on wideband time-of-arrival based localization,” *Proc. IEEE*, vol. 106, no. 7, pp. 1183–1203, 2018.
- [149] B. Peng and C. Papadimitriou, “The complexity of non-stationary reinforcement learning,” in *Proc. Int. Conf. Algorithmic Learn. Theory*, 2024, pp. 972–996.

- [150] L. Pucci and A. Giorgetti, “Position error bound for cooperative sensing in MIMO-OFDM networks,” *arXiv preprint arXiv:2405.19858*, 2024.
- [151] X. Wang, K. Tao, N. Cheng, Z. Yin, Z. Li, Y. Zhang, and X. Shen, “RadioDiff: An effective generative diffusion model for sampling-free dynamic radio map construction,” *IEEE Trans. Cogn. Commun. Netw.*, vol. 11, no. 2, pp. 738–750, 2025.
- [152] X. Wang, Q. Zhang, N. Cheng, R. Sun, Z. Li, S. Cui, and X. Shen, “Radiodiff- k^2 : Helmholtz equation informed generative diffusion model for multi-path aware radio map construction,” 2025. [Online]. Available: <https://arxiv.org/abs/2504.15623>
- [153] X. Wang, Z. Fang, N. Cheng, R. Sun, Z. Li, and X. Shen, “RadioDiff-Inverse: diffusion enhanced Bayesian inverse estimation for ISAC radio map construction,” 2025. [Online]. Available: <https://arxiv.org/abs/2504.14298>
- [154] H. Chen, F. Fang, and X. Wang, “Semantic extraction model selection for IoT devices in edge-assisted semantic communications,” *IEEE Commun. Lett.*, vol. 28, no. 7, pp. 1733–1737, 2024.
- [155] C. Liang, H. Du, Y. Sun, D. Niyato, J. Kang, D. Zhao, and M. A. Imran, “Generative ai-driven semantic communication networks: Architecture, technologies, and applications,” *IEEE Trans. Cogn. Commun. Netw.*, vol. 11, no. 1, pp. 27–47, 2025.
- [156] M. Angjelichinoski, D. Denkovski, V. Atanasovski, and L. Gavrilovska, “Cramér-rao lower bounds of RSS-based localization with anchor position uncertainty,” *IEEE Trans. Inf. Theory*, vol. 61, no. 5, pp. 2807–2834, 2015.
- [157] Z. Abu-Shaban, X. Zhou, T. Abhayapala, G. Seco-Granados, and H. Wymeersch, “Error bounds for uplink and downlink 3D localization in 5G millimeter wave systems,” *IEEE Trans. Wireless Commun.*, vol. 17, no. 8, pp. 4939–4954, 2018.

Appendices

Appendix A

Proof of Proposition 1

If the mobile device is scheduled for communication in the slot, we have $R = X_c^I R_0$; and otherwise, $R = 0$. As a result, the expected transmission rate for the device is

$$\mathbb{E}_{N_c^I} [R] = \rho_c \cdot \mathbb{E}_{N^I} [\mathbb{E}_{N_c^I} [X_c^I R_0 | N^I]] \quad (\text{A.1a})$$

$$\geq \rho_c \cdot \mathbb{E}_{N^I} \left[\frac{R_0 X_c^A}{1 + (N^I - 1)\rho_c} \right] \quad (\text{A.1b})$$

$$\geq \rho_c \cdot \frac{R_0 X_c^A}{1 + (\mathbb{E} [N^I] - 1)\rho_c}, \quad (\text{A.1c})$$

where (A.1b) holds since, based on Jensen's inequality, we have

$$\mathbb{E}_{N_c^I} [X_c^I | N^I] \geq \frac{X_c^A}{\mathbb{E}_{N_c^I} [N_c^I | N^I]} = \frac{X_c^A}{1 + (N^I - 1)\rho_c}, \quad (\text{A.2})$$

and (A.1c) holds also due to the Jensen's inequality.

Appendix B

Proof of Proposition 2

Under both the independent modeling and the joint modeling, the targets are modeled to be uniformly distributed in various directions around each mobile device. As a result, the directions of the beams formed by mobile devices in the sensing mode are independently and uniformly distributed in $[0, 2\pi)$ [55, 56]. In addition, since mobile devices independently access each spectrum band for the sensing with the probability $(1 - \rho_c)/X_s^I$, based on the property of PPPs, the mobile devices that access the same spectrum band for sensing form a thinned homogeneous PPP with the intensity $\check{\lambda}^I \cdot (1 - \rho_c)/X_s^I$. According to [55, Proposition 1], the CDF of the strongest interference I_s for the mobile device is :

$$\mathbb{P}\{I_s \leq i_s\} = \exp\left(-\frac{\check{\lambda}^I \cdot (1 - \rho_c)}{X_s^I} \cdot \frac{\phi_I^2}{4\pi} \cdot \left(\frac{P_s G_m^2 c^2}{(4\pi f_s)^2}\right) \cdot i_s^{-1}\right). \quad (\text{B.1})$$

Since $\gamma_s = P_e/I_s$, we have γ_s is $\mathbb{P}\{\gamma_s \geq \hat{\gamma}_s\} = \mathbb{P}\{I_s \leq \frac{P_e}{\hat{\gamma}_s}\}$, which should be greater than $\hat{P}$ to satisfy the sensing quality requirement. Therefore, we have

$$P_e \geq \hat{\gamma}_s \left(-\ln \hat{P}\right)^{-1} \cdot \left(\frac{\phi_I^2(1 - \rho_c)\lambda_1^I}{4\pi X_s^I}\right) \cdot \frac{P_s G_m^2 c^2}{(4\pi f_s)^2}. \quad (\text{B.2})$$

The power received sensing signals, i.e., P_e , is negatively correlated with the distance between the mobile device and the target, i.e., D . As a result, to satisfy (B.2), there exists a maximum value of the distance. By substituting P_e in (B.2) with its calculation formula, the maximum distance can be calculated as the right-hand side of (3.10).

Consider $\Phi = \{\rho_c, X_c^A, X_s^I, X_s^A, F_e^A, D_{\max}^{I,U}\}$ as one of the optimal solutions if there exist more than one optimal solutions for Problem (P1). According to the proof in the first

step, $D_{\max}^{\text{I,U}}$ should be equal to or smaller than the term on the right-hand side of (3.10) to make constraint (3.8g) in Problem (P1) satisfied. As a result, we consider the following two cases: (i) In Case 1, $D_{\max}^{\text{I,U}}$ is equal to the term on the right-hand side of (3.10). In this case, the considered optimal solution makes (3.10) satisfied; (ii) In Case 2, $D_{\max}^{\text{I,U}}$ is smaller than the term on the right-hand side of (3.10). We can thus increase $D_{\max}^{\text{I,U}}$ to equal the term on the right-hand side of (3.10), which does not impact the overall resource consumption and ensures that all constraints in Problem (P1) remain satisfied. Accordingly, the solution after the increase of $D_{\max}^{\text{I,U}}$ is still optimal and makes (3.10) satisfied. If there exists one unique optimal solution for Problem (P1), for the optimal solution, $D_{\max}^{\text{I,U}}$ must be equal to the term on the right-hand side of (3.10), i.e., making (3.10) satisfied. This is because, otherwise, according to Case 2, we can find another optimal solution for (P1).

Appendix C

Proof of Proposition 3

We respectively prove the three necessary conditions in (3.11). First, defined in (3.7a), an upper bound of the expected number of targets that can be sensed by the mobile device is

$$\bar{N}_I^U = \min \left\{ \mathbb{P} \{ D^{I,U} \leq D_{\max}^{I,U} \} \cdot \mathbb{E} [N_{I,1}^U], \frac{1 - \rho_c}{\hat{\rho}_s}, \frac{\tau T \cdot F_e^I}{C_0} \right\}, \quad (C.1)$$

As a result, if $(1 - \rho_c)/\hat{\rho}_s$ is greater than $\mathbb{E}[N_{I,1}^U]$ or $\tau T \cdot F_e^I/C_0$, we can find a greater ρ_c which decreases the amount of required spectrum resources, i.e., X_c^A , while keeping all the other parts of Problem (P1), including $\bar{N}_I^U$, unaffected. Therefore, for the optimal solutions to Problem (P1), ρ_c should satisfy the condition in (3.11a):

$$\frac{1 - \rho_c}{\hat{\rho}_s} \leq \min \left\{ \mathbb{E} [N_{I,1}^U], \frac{\tau T \cdot F_e^I}{C_0} \right\}. \quad (C.2)$$

Second, with the necessary condition, (C.1) reduces to

$$\bar{N}_I^U = \min \left\{ \mathbb{P} \{ D^{I,U} \leq D_{\max}^{I,U} \} \cdot \mathbb{E} [N_{I,1}^U], \frac{1 - \rho_c}{\hat{\rho}_s} \right\}. \quad (C.3)$$

According to the expression derived in Section 3.4.1, $\mathbb{P} \{ D^{I,U} \leq D_{\max}^{I,U} \}$ increases with $D_{\max}^{I,U}$ which increases with X_s^I and decreases with ρ_c , and $\mathbb{E} [N_{I,1}^U]$ is not affected by any decision variables. As a result, first, if $\mathbb{P} \{ D^{I,U} \leq D_{\max}^{I,U} \} \cdot \mathbb{E} [N_{I,1}^U] > (1 - \rho_c)/\hat{\rho}_s$, we can find a smaller X_s^I while keeping all the other parts of Problem (P1) including $\bar{N}_I^U$ unaffected. Second, if $\mathbb{P} \{ D^{I,U} \leq D_{\max}^{I,U} \} \cdot \mathbb{E} [N_{I,1}^U] < (1 - \rho_c)/\hat{\rho}_s$, we can find a greater ρ_c which (i) increases $\bar{N}_I^U$

such as to decrease the amount of spectrum and computing resources required for target tracking by the APs; (ii) decreases the amount of required spectrum resources, i.e., X_c^A . To conclude, for the optimal solutions, the condition in (3.11b) should be satisfied:

$$\frac{1 - \rho_c}{\hat{\rho}_s} = \mathbb{P} \{ D^{I,U} \leq D_{\max}^{I,U} \} \cdot \mathbb{E} [N_{I,1}^U]. \quad (\text{C.4})$$

Third, for the necessary condition (3.11c), if it is violated, we can find a smaller amount of required spectrum resources, i.e., X_s^A , or computing resources, i.e., F_e^A , without affecting all the other parts of Problem (P1).

Appendix D

Proof of Proposition 4

Case 1: If $T_{n-1}^{\text{lc}} + T_{n-1}^{\text{lc}} \leq \Delta T_{n-1}$, the on-device queuing delay of the n -th DNN task T_n^{lc} equals to zero. Based on (4.15), $D_{n-1 \rightarrow n}^{\text{lc}} = \max\{\min\{T_{n-1}^{\text{lc}} - \Delta T_{n-1}, 0\} + T_{n-1}^{\text{lc}}, 0\} = 0$. Similarly, we have $D_{m \rightarrow n}^{\text{lc}} = 0, \forall m < n$, we have $D_{m \rightarrow n}^{\text{lc}} = 0, \forall m \geq n$. Therefore, $T_n^{\text{lc}} = \sum_{m=1}^N D_{m \rightarrow n}^{\text{lc}}$.

Case 2: If $T_{n-1}^{\text{lc}} + T_{n-1}^{\text{lc}} > \Delta T_{n-1}$, $T_n^{\text{lc}} = T_{n-1}^{\text{lc}} + T_{n-1}^{\text{lc}} - \Delta T_{n-1} > 0$, which indicates that one preceding DNN task (its index is denoted by m_0) is in the computing unit upon the generation of the n -th task. As a result, we have:

$$T_{m_0}^{\text{lc}} \leq \sum_{i=m_0}^{n-1} \Delta T_i \leq T_{m_0}^{\text{lc}} + T_{m_0}^{\text{lc}}, \quad (\text{D.1})$$

and for any m -th task, where $m_0 + 1 \leq m \leq n - 1$, $T_m^{\text{lc}} \geq \sum_{i=m}^{n-1} \Delta T_i$.

Based on (4.15), $D_{m \rightarrow n}^{\text{lc}}$ can be calculated by:

$$D_{m \rightarrow n}^{\text{lc}} = \begin{cases} T_m^{\text{lc}} + T_m^{\text{lc}} - \sum_{i=m}^{n-1} \Delta T_i, & m = m_0, \\ T_m^{\text{lc}}, & m_0 + 1 \leq m \leq n - 1, \\ 0, & \text{otherwise.} \end{cases} \quad (\text{D.2})$$

Then, $\sum_{n=1}^N D_{m \rightarrow n}^{\text{lq}}$ can be calculated as:

$$\begin{aligned}
& \sum_{n=1}^N D_{m \rightarrow n}^{\text{lq}} \\
&= T_{m_0}^{\text{lq}} + T_{m_0}^{\text{lc}} - \sum_{i=m_0}^{n-1} \Delta T_i + \sum_{m=m_0+1}^{n-1} T_m^{\text{lc}} \\
&= (T_{m_0}^{\text{lq}} + T_{m_0}^{\text{lc}} - \Delta T_{m_0}) - \sum_{i=m_0+1}^{n-1} \Delta T_i + \sum_{m=m_0+1}^{n-1} T_m^{\text{lc}} \\
&= T_{m_0+1}^{\text{lq}} - \sum_{i=m_0+1}^{n-1} \Delta T_i + \sum_{m=m_0+1}^{n-1} T_m^{\text{lc}} \\
&= T_{m_0+1}^{\text{lq}} + T_{m_0+1}^{\text{lc}} - \sum_{i=m_0+1}^{n-1} \Delta T_i + \sum_{m=m_0+2}^{n-1} T_m^{\text{lc}} \\
&\dots \\
&= T_{n-1}^{\text{lq}} + T_{n-1}^{\text{lc}} - \Delta T_{n-1} \\
&= T_n^{\text{lq}}.
\end{aligned} \tag{D.3}$$

Appendix E

Proof of Proposition 5

When $T_n^{\text{lc}} = 0$, $D_{n \rightarrow m}^{\text{lc}} = 0, \forall 1 \leq m \leq N$. As a result, $D_n^{\text{lc}} = \sum_{m=1}^N D_{n \rightarrow m}^{\text{lc}} = 0$. When $T_n^{\text{lc}} \geq \Delta T$, for brevity, we denote $Q^D(t_{n,0})$ and $Q^D(t_{n,0} + (T_n^{\text{lc}}/\Delta T) - 1)$ by q_0 and q_1 , which represent the number of tasks in the on-device task queue in the time slots when the on-device inference for the n -th DNN task starts and completes, respectively. During the on-device inference for the n -th task, the m -th task is in the on-device task queue, where $n < m \leq n + q_1$. We can derive the queuing delay of the m -th task that results from the on-device inference of the n -th task as:

$$D_{n \rightarrow m}^{\text{lc}} = \begin{cases} T_n^{\text{lc}}, & n < m \leq n + q_0, \\ T_n^{\text{lc}} + T_n^{\text{lc}} - \sum_{i=n}^{m-1} \Delta T_i, & n + q_0 + 1 \leq m \leq n + q_1, \end{cases} \quad (\text{E.1})$$

and calculate on-device workload during the on-device inference of the n -th task by:

$$Q^D(t) = \begin{cases} q_0, & t_{n,0} \leq t < \frac{1}{\Delta T} \sum_{i=1}^{n+q_0} \Delta T_i, \\ m-n, & \frac{1}{\Delta T} \sum_{i=1}^{n+q_0} \Delta T_i \leq t < \frac{1}{\Delta T} \sum_{i=1}^{m-1} \Delta T_i, \\ q_1, & \frac{1}{\Delta T} \sum_{i=1}^{n+q_1-1} \Delta T_i \leq t < t_{n,0} + \frac{1}{\Delta T} T_n^{\text{lc}}, \end{cases} \quad (\text{E.2})$$

where $q_0 + n \leq m \leq q_1 + n$. Based on (E.1) and (E.2), we can calculate $\sum_{m=1}^N D_{n \rightarrow m}^{\text{lc}}$ and $\sum_{t=t_{n,0}}^{t_{n,0} + (T_n^{\text{lc}}/\Delta T) - 1} Q^D(t) \Delta T$ using (E.3) and (E.4), respectively, and it can be observed from (E.3) and (E.4) that they are equal.

$$\begin{aligned}
D_n^{\text{Iq}} &= \sum_{m=1}^N D_{n \rightarrow m}^{\text{Iq}} \\
&= q_0 T_n^{\text{Ic}} + \sum_{m=n+q_0+1}^{n+q_1} (T_n^{\text{Iq}} + T_n^{\text{Ic}} - \sum_{i=n}^{m-1} \Delta T_i) \\
&= (q_1 - q_0) T_n^{\text{Iq}} + q_1 T_n^{\text{Ic}} - (q_1 - q_0) \sum_{i=n}^{n+q_0} \Delta T_i - \sum_{i=q_0+n+1}^{q_1+n-1} (q_1 - i + n) \Delta T_i
\end{aligned} \tag{E.3}$$

$$\begin{aligned}
&\sum_{t=t_{n,0}}^{t_{n,0}+(T_n^{\text{Ic}}/\Delta T)-1} Q^D(t) \Delta T \\
&= q_0 (-t_{n,0} \Delta T + \sum_{i=1}^{n+q_0} \Delta T_i) + \sum_{m=q_0+n+1}^{q_1+n-1} (m-n) \Delta T_m + q_1 (t_{n,0} \Delta T + T_n^{\text{Ic}} - \sum_{i=1}^{n+q_1-1} \Delta T_i) \\
&= q_0 (-T_n^{\text{Iq}} + \sum_{i=n}^{n+q_0} \Delta T_i) + \sum_{m=q_0+n+1}^{q_1+n-1} (m-n) \Delta T_m + q_1 (T_n^{\text{Iq}} + T_n^{\text{Ic}} - \sum_{i=n}^{n+q_1-1} \Delta T_i) \\
&= (q_1 - q_0) T_n^{\text{Iq}} + q_1 T_n^{\text{Ic}} - (q_1 - q_0) \sum_{i=n}^{n+q_0} \Delta T_i + (\sum_{m=q_0+n+1}^{q_1+n-1} (m-n) \Delta T_m - q_1 \sum_{i=q_0+n+1}^{q_1+n-1} \Delta T_i) \\
&= (q_1 - q_0) T_n^{\text{Iq}} + q_1 T_n^{\text{Ic}} - (q_1 - q_0) \sum_{i=n}^{n+q_0} \Delta T_i - \sum_{i=q_0+n+1}^{q_1+n-1} (q_1 - i + n) \Delta T_i
\end{aligned} \tag{E.4}$$

Appendix F

Proof of Proposition 7

We divide the observations in t -th frame into two sets, and derive the EFIMs of the AR device position given each set of observations, respectively.

The first set includes (i) the estimated channel parameters $[\hat{\tau}_0^{(t)}, \dots, \hat{\tau}_K^{(t)}]^T$ that are modeled as Gaussian distribution with mean $[\tau_0^{(t)}, \dots, \tau_K^{(t)}]^T$ and covariance matrix $\text{diag}(\tilde{\varepsilon}_{\tau_0}^{(t)}, \dots, \tilde{\varepsilon}_{\tau_K}^{(t)})$; (ii) the position of the anchors in the global environment map that follow Gaussian distribution with mean vector $[\mathbf{q}_0^T, \dots, \mathbf{q}_K^T]^T$ and covariance matrix $\text{diag}(\bar{\varepsilon}_{q_0}^{(t)}, \bar{\varepsilon}_{q_0}^{(t)}, \dots, \bar{\varepsilon}_{q_K}^{(t)}, \bar{\varepsilon}_{q_K}^{(t)})$. According to [156, Theorem 1], the EFIM of AR device position given this set of observations can be derived as $\sum_{k=0}^K \lambda_k^{(t)} (\mathbf{v}_k^{(t)})^T \mathbf{v}_k^{(t)}$.

The second set includes (i) the estimated channel parameters $[\varepsilon_{\theta_{R,0}}^{(t)}, \dots, \varepsilon_{\theta_{R,K}}^{(t)}]^T$ that are modeled as Gaussian distribution with mean vector $[\tilde{\varepsilon}_{\theta_{R,0}}^{(t)}, \dots, \tilde{\varepsilon}_{\theta_{R,K}}^{(t)}]^T$ and covariance matrix $\text{diag}(\tilde{\varepsilon}_{\tau_0}^{(t)}, \dots, \tilde{\varepsilon}_{\tau_K}^{(t)})$; (ii) the position of the anchors in the environment map that follow Gaussian distribution with mean $[\mathbf{q}_0^T, \dots, \mathbf{q}_K^T]^T$ and covariance matrix $\text{diag}(\bar{\varepsilon}_{q_0}^{(t)}, \bar{\varepsilon}_{q_0}^{(t)}, \dots, \bar{\varepsilon}_{q_K}^{(t)}, \bar{\varepsilon}_{q_K}^{(t)})$. Similarly, according to [156, Theorem 1], the EFIM of the AR device given this set of observations can be derived as $\sum_{k=0}^K \mu_k^{(t)} (\mathbf{w}_k^{(t)})^T \mathbf{w}_k^{(t)}$.

Consider that the two sets of observations are independent. Due to the addition property of Fisher information, the EFIM of the AR user position given the two sets of observations can be approximated as the addition of the two EFIMs given each set of observations. The average squared positioning error in each coordinate axis is then obtained as (5.2).

Appendix G

Proof of Proposition 8

Similar to [157], considering the independence between the channel parameters of the two paths, i.e., $[\Theta_0^{(t)}, \Theta_k^{(t)}]$, $\forall k = 1, 2, \dots, K$, the EFIM of them is denoted by $\mathcal{I}_k^{(t)}$ and written as

$$\mathcal{I}_k^{(t)} = \begin{bmatrix} \mathbf{J}_0^{(t)} & \mathbf{0} \\ \mathbf{0} & \mathbf{J}_k^{(t)} \end{bmatrix}, \quad (\text{G.1})$$

where $\mathbf{J}_0^{(t)}$ and $\mathbf{J}_k^{(t)}$ can be calculated according to (5.1). Given the channel parameters from the two paths, i.e., $[\Theta_0^{(t)}, \Theta_k^{(t)}]$, the EFIM of the position of the AR device and the anchor, i.e., $[\mathbf{p}_t, \mathbf{q}_k]$, is denoted by $\mathcal{K}_k^{(t)}$ and calculated as

$$\mathcal{K}_k^{(t)} = \mathcal{M}_k^{(t)\text{T}} \mathcal{I}_k^{(t)} \mathcal{M}_k^{(t)}, \quad (\text{G.2})$$

where $\mathcal{M}_k^{(t)}$ is the Jacobian matrix that is given by

$$\mathcal{M}_k^{(t)} = \begin{bmatrix} \frac{\partial \Theta_0^{(t)}}{\partial \mathbf{p}_t} & \frac{\partial \Theta_0^{(t)}}{\partial \mathbf{q}_k} \\ \frac{\partial \Theta_k^{(t)}}{\partial \mathbf{p}_t} & \frac{\partial \Theta_k^{(t)}}{\partial \mathbf{q}_k} \end{bmatrix}, \quad (\text{G.3})$$

and the submatrices, $\forall k' \in \{0, k\}$, are defined as

$$\frac{\partial \Theta_{k'}^{(t)}}{\partial \mathbf{p}_t} \triangleq \begin{bmatrix} \frac{\partial \tau_{k'}^{(t)}}{\partial p_{t,x}} & \frac{\partial \tau_{k'}^{(t)}}{\partial p_{t,y}} \\ \frac{\partial \theta_{\text{R},k'}^{(t)}}{\partial p_{t,x}} & \frac{\partial \theta_{\text{R},k'}^{(t)}}{\partial p_{t,y}} \end{bmatrix}, \quad \frac{\partial \Theta_{k'}^{(t)}}{\partial \mathbf{q}_k} \triangleq \begin{bmatrix} \frac{\partial \tau_{k'}^{(t)}}{\partial q_{k,x}} & \frac{\partial \tau_{k'}^{(t)}}{\partial q_{k,y}} \\ \frac{\partial \theta_{\text{R},k'}^{(t)}}{\partial q_{k,x}} & \frac{\partial \theta_{\text{R},k'}^{(t)}}{\partial q_{k,y}} \end{bmatrix}. \quad (\text{G.4})$$

According to (5.3), the entries of the submatrices $\frac{\partial \Theta_{k'}^{(t)}}{\partial \mathbf{p}_t}, \forall k' \in \{0, k\}$, can be calculated as

$$\frac{\partial \tau_{k'}^{(t)}}{\partial p_{t,x}} = \frac{1}{c} \cos \theta_{T,k'}, \quad \frac{\partial \tau_{k'}^{(t)}}{\partial p_{t,y}} = \frac{1}{c} \sin \theta_{T,k'}, \quad (\text{G.5a})$$

$$\frac{\partial \theta_{R,k'}^{(t)}}{\partial p_{t,x}} = -\frac{\sin \theta_{T,k'}}{\|\mathbf{p}_t - \mathbf{q}_{k'}\|_2}, \quad \frac{\partial \theta_{R,k'}^{(t)}}{\partial p_{t,y}} = \frac{\cos \theta_{T,k'}}{\|\mathbf{p}_t - \mathbf{q}_{k'}\|_2}. \quad (\text{G.5b})$$

The entries of the submatrix $\frac{\partial \Theta_k^{(t)}}{\partial \mathbf{q}_k}$ can be calculated as

$$\frac{\partial \tau_k^{(t)}}{\partial q_{k,x}} = -\frac{1}{c} \cos \theta_{T,k}, \quad \frac{\partial \tau_k^{(t)}}{\partial q_{k,y}} = \frac{1}{c} \sin \theta_{T,k}, \quad (\text{G.6a})$$

$$\frac{\partial \theta_{R,k}^{(t)}}{\partial q_{k,x}} = \frac{\sin \theta_{T,k}}{\|\mathbf{p}_t - \mathbf{q}_k\|_2}, \quad \frac{\partial \theta_{R,k}^{(t)}}{\partial q_{k,y}} = -\frac{\cos \theta_{T,k}}{\|\mathbf{p}_t - \mathbf{q}_k\|_2}, \quad (\text{G.6b})$$

and the entries of the submatrix $\frac{\partial \Theta_1^{(t)}}{\partial \mathbf{q}_k}$ are all zeros since the channel parameters in the first path do not depend on the position of the k -th VA.

The position error of the k -th VA, $\varepsilon_k^{(t)}$, is bounded as

$$\varepsilon_k^{(t)} \geq \text{tr} \left(\left[\boldsymbol{\kappa}_k^{(t)} \right]_{3:4,3:4}^{-1} \right), \quad (\text{G.7})$$

where the right-hand side can be calculated as the right-hand side of (5.4) through a lengthy calculation.