A Study on Intelligent Resource Allocation Based on Deep Learning for Internet of Thing

A dissertation presented
by

Fengxiao Tang

submitted to
Tohoku University
in partial fulfillment of the requirements
for the degree of

Doctor of Philosophy

Supervisor: Professor Nei Kato

Department of Applied Information Sciences
Graduate School of Information Sciences
Tohoku University

January, 2019

A Study on Intelligent Resource Allocation Based on Deep Learning for Internet of Thing

A dissertation presented by

Fengxiao Tang

approved as to style and content by

Professor Nei Kato,
Graduate School of Information Sciences

Professor Xiao Zhou,
Graduate School of Information Sciences

Professor Takuo Suganuma,
Graduate School of Information Sciences

Associate Professor Zubair Md. Fadlullah,
Graduate School of Information Sciences

Tohoku University
Sendai, Japan.

To My Family

Abstract

Due to the fast increase of sensing data and quick response requirement in the Internet of Things (IoT), the large amount of network traffic flows and computing tasks become too heavy to be supported by the limited radio and traffic resources in the IoT. In this study, we analysis the resource limitation in the IoT, three corresponding resource allocation algorithms are proposed to manage the limited radio and traffic resources in IoT.

In the first proposal, the UAV-enabled IoT is proposed as the resource management platform. In the UAV-enabled IoT, the UAV-mounted flying base stations using both D2D and cellular connection is deployed as high dynamic edge computing server to provide content sharing and delivery service to both IoT devices and central cloud server. Based on the UAV-enabled IoT, a anti-cooperation game based partially channel assignment (POC) algorithm referred to as ACPOCA is proposed to dynamically allocate POC to each link in the IoT. In our proposed game theory based AC-POCA, the device use only local information to play the game, and reach a steady state, uniqueness of which is verified through analysis. Also, the upper bound of AC-POCA (i.e., Price of Anarchy) is analytically evaluated, which is corroborated by simulation results. In addition, simulation results demonstrate the effectiveness of AC-POCA in terms of good throughput and low signaling overhead in a dynamic network environment.

After the AC-POCA is proposed, we further consider the radio resource allocation in the more real IoT environment with high dynamic changed traffic flows such as bursty traffic in the network. Such kind of high dynamics of traffic load make the conventional fixed channel assignment based radio allocation algorithm ineffective. Furthermore, consider the tremendous number of devices using various underlying protocols to connect in IoT. The Software Defined Networking (SDN) based IoT referred to as SDN-IoT is considered to deal with the heterogeneous resources and underlying protocols of IoT. In the vein, a Deep Learning based Partially Channel Assignment Algorithm, referred to as DLPOCA, is proposed to intelligently allocate channels to each link in the SDN-IoT network. In addition, to deal with the high dynamic bursty traffic, we further consider a deep learning based prediction method to estimate the future traffic in IoT. Then, the traffic prediction based novel intelligent channel assignment algorithm (TP-DLPOCA) is proposed, which can intelligently avoid potential congestion and quickly assign suitable channels in SDN-IoT. The simulation result demonstrates that our proposal significantly outperforms conventional channel assignment algorithms and ACPOCA in the high dynamic network environment.

Finally, After the suitable radio resources are intelligently allocated to the links in the IoT, a deep learning based network traffic allocation algorithm is proposed to further optimize the network flows allocation in the IoT. In the proposal, we propose appropriate input and output characterizations of heterogeneous network traffic and propose a online deep neural networks system. We describe how our proposed deep learning based network traffic control system works and differs from traditional neural networks. With the proposal, the network traffic can be intelligent allocated to suitable routing path, which can continuously learn from historical experiences to improve the allocation decision by itself. Also, simulation results are reported which demonstrate the encouraging performance of our proposed deep learning system compared to a benchmark routing strategies in terms of significantly better signaling overhead, throughput, and delay.

Acknowledgments

First and foremost, I would like to show my deepest gratitude to my supervisor, Prof. Nei Kato, for his continuous guidance, supervision, and warm support during my study in lab. His kindness, prudence and work of ethics have made my PhD period one of the best periods of my life. I am also grateful to my supervisor Dr. Zubair Md. Fadlullah, He is so kind and warm to guide me of both my essential research direction and fundamental in the PHD period.

Second, In particular, I would like to show my deepest gratitude to my family. Thanks for the selfless support from my parents and grandfather, also thanks for the helpful advises and guideline from my dear uncle.

This dissertation would not be possible without the helpful scholarship form the CSC. Grate Respect for their strong support that enabled me to pursue the challenging route during my doctoral years. Also give grate thankful to my college Bomin Mao and other co-others who cooperates with me during the research project.

Last but not least, I' d like to thank all my friends, especially my roommate Yu Fang, for their encouragement and support.

Contents

List of Figures

List of Tables

List of Acronyms

AC-POCA Anti-Coordination based Partially Overlapping Channel Assignment

AP Access Point

CNNs Convolutional Neural Networks

C-RAN(CRAN) Centralized, Cloud Computing-based Architecture For Radio Access Networks

CRNs cognitive radio networks

CC Common Channel

CTP Traffic Load Prediction

CoCAG Cooperative Channel Assignment Game

CPU Central Processing Unit

D2D Device-to-Device

DBMs Deep Boltzmann Machines

DBA Deep Belief Architecture

DTP Distributed control Traffic load Prediction

DBN Deep Belief Network

DLPOCA Deep Learning based Partially Overlapping Channel Assignment

EIC Event Intensive Case

FDMA Frequency Division Multiple Access

GPU Graphics processing unit

GW Gateway

HC	Hop Count
IS-IS	Intermediate System to Intermediate System
IF	Interference Factor
IEEE	Institute of Electrical & Electronics Engineers
IoT	Internet of Things
LSAs	link-state advertisements
MRMC	Inter-Packet-Delay
MRMC	Multi-Radio Multi-Channel
MRF	Markov Random Field
M2M	Machine to Machine
MEC	Mobile-Edge Computing
NFV	Distributed Dynamic Resource Allocation
NFC	Near Field Communication
NICs	Network interface controllers
NE	Nash Equilibrium
NI	Non-Interfering
OSPF	Open Shortest Path Firs
OC	Overlapping Channel
PoA	Price of Anarchy
POCs	Partially Overlapping Channels
PIC	Periodic Intensive Case
QoE	Quality of Experience
QoS	Quality of Service
RIP	Routing Information Protocol
RFID	Radio Frequency Identication

SDN	Software Defined Networking
SDN-IoT	SDN enabled IoT
SP	Shortest Path
SDRs	Software Defined Routers
S-CTP	Semi-Central control Traffic load Prediction
SINR	Signal-to-Interference-plus Noise Ratio
TP-DLPOCA	Combined Traffic Prediction and DLPOCA
TL	Traffic Load
UAVs	Unmanned Aerial Vehicles
WLANs	Wireless Local Area Networks
WMN	Wireless Mesh Network
WM	Weight Matrix
WSN	Wireless Sensor Networks

Chapter 1

Introduction

1.1 Background-The development and status of IoT

With the wide use of high speed internet, the physical objects embedded with electronics, software, sensors, actuators can easily connect to internet and construct as a big smart things driven network. Those smart network is commonly known as Internet of things (IoT) [1, 2, 3, 4] , and the physical objects are referred to as IoT devices. With the rapid development of internet, there are more than 200 billion IoT devices is expected to interconnected around the world in the next 3 years. [5]

With the IoT paradigm, the researchers envisions the scenario contains a large number of smart physical objects equipped with micro controllers, transceivers to collect data and communication with each other in our daily life. It is foreseeable that, those intelligent networks will integrate into our life and become an integral part of the Internet. [6]

As the rapid increase of IoT devices, there is an associated business market for both manusfacturers, network services providers, software developers and platform builders. The IoT devices and corresponding equipments are expected to reach 212 billions around 2020 [7]. Both the traffic flows and connected machines are grows fast in the past 10 years [8, 9], especially the M2M devices increased more than three times during 2017-2012 [10]. The applications of IoT are widely distributed in various areas, such as health-care, industry management, automation systems. In addition, the Navigant research report Shows global revenue from IoT and analytics for utilities market may Grow to $5.1 billion in 2028 [11]. moreover, the market of IoT application in smart city is estimated to continuous increase for nearly 16 billions per year in the future years. [12, 6]

The rapid growth of IoT in both industry and business fields emerges a big challenge and opportunity for researchers. The traditional network structures only consider the simple server and clients are no longer suitable for the new IoT system. the traditional Internet architecture needs to be revised to match the IoT challenges. For example, the tremendous number of objects swilling to connect to the Internet should be considered in many underlying protocols [5]. IoT draws together various technologies such as Radio Frequency Identication (RFID), Near Field Communication (NFC), Wireless Sensor Networks (WSN), Machine-to-Machine (M2M), Unmanned Aerial Vehicles (UAVs) communications. The complexity, possible limitations and heterogeneity of various IoT devices connected to the internet will require even more specic tools to manage them and to improve the performance of the whole network [13]. In such complex IoT environment, the requirement of communication QoS of devices is tremendously increasing with both the computational resource and radio resource of devices in IoT is limited. How to balance the increased QoS requirement and limited resources becomes critical problem.

1.2 Introduction of the resource allocation in IoT

The resource allocation algorithms are widely researched in traditional network, such as channel allocation in cellular network [14], network traffic allocation in wireless mesh network [15], computing resource offloading in mobile network [16] and energy resource allocation in WSN [17]. In the thesis, we mainly focus on the channel allocation and network traffic allocation.

1.2.1 Considered UAV-anabled IoT and Anti-Coordination Game based dynamic POC Assignment algorithm (ACPOCA)

The cloud computing, as an novel internet-based computing platform, can provides shared processing resources computers and other devices on demand [16]. In traditional network, the cloud server is normally deployed as the center service to offload computing resources form distributed devices. Recently, the mobile-edge computing (MEC) has gained momentum to expend the resource offloading tasks from the center cloud computing to the edge nodes in IoT [18]. Moreover, as another hot technique, Unmanned Aerial Vehicles (UAVs) have appeared as a promising candidate to be exploited as flying base stations [19] to quickly construct efficient and high Quality-of-Service (QoS) wireless networks even in remote and/or rural areas [20, 21]. Such kind of flying base stations can be deployed as high dynamic edge computing service provider in both traditional cellular network and the considered IoT. On the other hand, in IoT, the bandwidth and energy are limited at the content servers when multiple users aim to access the same content. This limitation results in increased resource waste and delay [22]. To address this issue, Device-to-Device (D2D) [23, 24, 25, 26] communication with caching emerged as a complementary solution in IoT [27]. D2D communication reuses existing licensed spectrum resources to make under-laid transmission links, which can typically be deployed between smart devices [28], which is a widely used technique in IoT. Furthermore, UAVs, to exploit their earlier

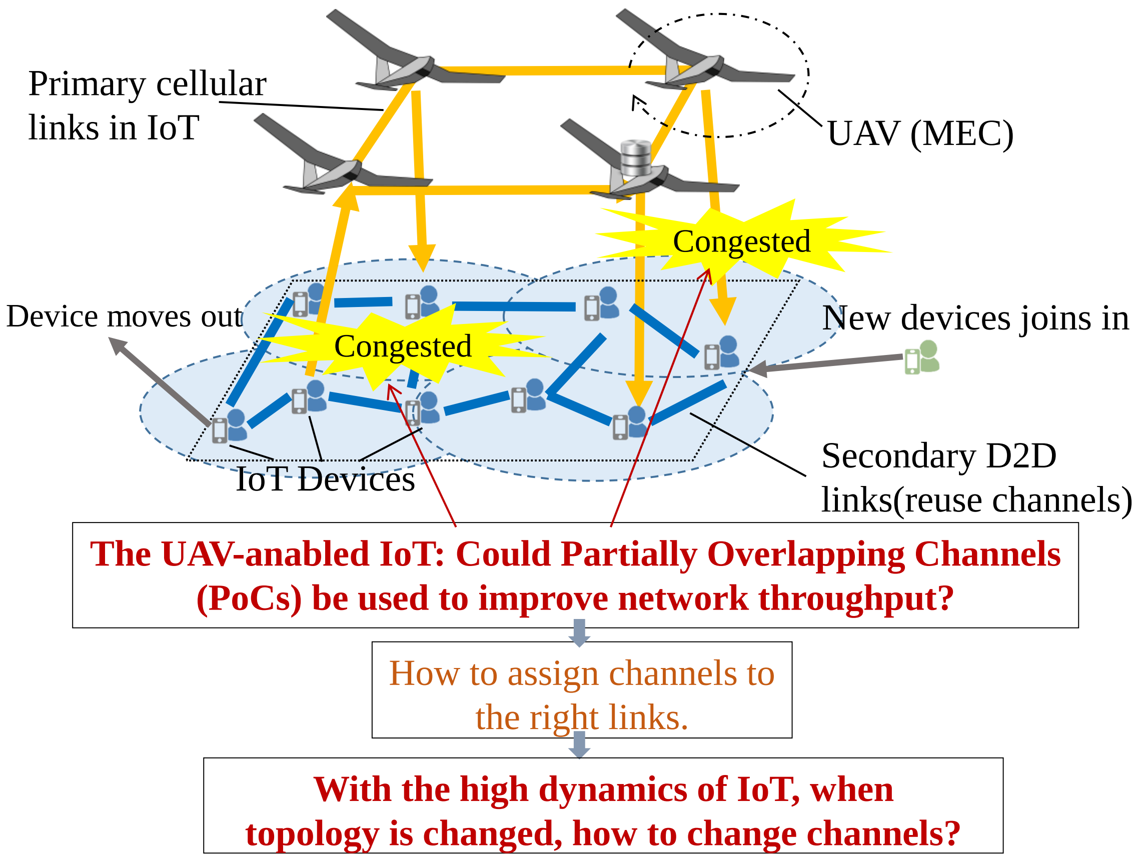

Primary cellular links in IoT
UAV (MEC)
Congested
Device moves out
Congested
New devices joins in
IoT Devices
Secondary D2D links(reuse channels)
The UAV-anabled IoT: Could Partially Overlapping Channels (PoCs) be used to improve network throughput?
How to assign channels to the right links.
With the high dynamics of IoT, when topology is changed, how to change channels?

Figure 1.1: Considered UAV-enabled IoT architecture and the problem of using partially overlapping channels in the network.

mentioned flexibility to construct wireless communication networks in locations lacking adequate cellular infrastructure, have been used to establish D2D links with caching [20]. With such UAV-anabled D2D, the whole UAV base stations based MEC in IoT is referred to as UAV-enabled IoT. With the high dynamical ability of UAV, the infrastructure of the UAV-enabled IoT is not only used to offload computing but also can be flexibly changed to form a dynamic wireless network provider to meet the varying IoT user requirements.

In this research, we first envision the UAV-enabled IoT whereby the primary links (i.e., downlink transmission) and secondary links (i.e., D2D underlink communication) are considered as complementary methods for content delivery to offer better performance compared to conventional content delivery approaches.

However, as demonstrated in Fig.1.1 in the proposed combined heterogeneous network, both primary and D2D links share the same spectrum whereby both the UAVs and IoT devices typically use multi-radio, multi-channel communications. The spectrum used by both primary and secondary links of the UAV-enabled IoT makes the channel resources limited in the entire network, and the nearby channels easily become overlapping. Such overlapping channels in the neighboring UAVs and IoT devices can cause severe interference leading to network congestion. Since high interference exists between non-orthogonal (overlapping) channels and the number of orthogonal channels is limited, the Partially Overlapping Channel (POC) can be a good solution to decrease interference and improve network throughput [29, 30, 31] .To maximize the channel resource utilization, POCs can be assigned to the UAVs and IoT devices. Researches demonstrated that proper assignment of POCs can efficiently avoid interference and improve the aggregate throughput of various communication networks. However, current POC algorithms mostly focus on the improvement of network performance after channel assignment, but lack the consideration of waste throughput due to the suspended transmission during the channel assignment process. With the high dynamics of the considered IoT, the assigned channels need to be frequently changed to adaptively adjust to the dynamically changed network traffic. This dynamic adjustment throws out a critical requirement for the quick processing of the channel assignment. Therefore, How to efficiently assign POCs to the nodes while minimizing interference is a critical problem. Based on the conventional channel assignment problem, by considering the mobility of UAVs and IoT devices, the network topology becomes highly dynamic, which means that the link state and interference range of each node may frequently change. However, conventional POCs assignment are typically limited by complex and numerous iterations, which depends on the persistent global information and cause significantly long convergence time. Therefore, the existing POCs assignment in other communication networks may not applicable to the highly dynamic environment in the considered UAV-enabled IoT. This poses a further challenge to the channel assignment problem. In the first part of our research, based on the complex conditions in UAV-enabled IoT, I address those issues, and propose an Anti-Coordination Game [32] based dynamic POC Assignment algorithm, referred to as AC-POCA.

The contributions of the first part are as follows.

- I present a UAV-anabled IoT and justify the adopted network topology. Then, I analyze the new features of channels assignment problems in the proposed network.
- According to the new features of the proposed network, I use the anti-coordination game to model the channel assignment problem in the considered network that uses local information and leads to quick convergence time.
- Based on the high mobility of the UAV-enabled IoT, the dynamic topology based POC assignment algorithm is further designed to deal with situations in which the network environment is dynamically changed.
- I prove the existence and uniqueness of the steady state in AC-POCA and demonstrate its superior performance over comparable methods in both mixed and dynamic environments.

1.2.2 Considered SDN-IoT and proposed Deep Learning based Partially Overlapping Channel Assignment (DLPOCA)

As the mentioned in the background, a big challenge of IoT is that the tremendous number of objects swilling to connect to the Internet should be considered in many underlying protocols. If all IoT devices and corresponding communications are constructed in the same structure and protocols, the resources algorithms maybe easily designed, however, the real world IoT deployments are fundamentally heterogeneous. Software Defined Networking (SDN) [33, 34] is a famous technique used in the IoT to deal with the heterogeneous resources and structure [35] . In such SDN-IoT as depicted in Fig. 1.2, heterogeneous devices sense and collect data in the sensing plane, and then send the data to the gateway after integration through switches (i.e., routers) in the data plane. With SDN, the software enabled with resource allocation algorithm are commonly deployed upon the sensing plane of heterogeneous sensor devices. In such structure, the different underlying protocols no longer the bottleneck of designed resource allocation algorithm. However, with the increasing number of devices, the load of integrated traffic in switches may become significantly heavy, and multiple radio channels are needed to be evenly assigned to each link to balance the load [36, 37, 38].

To solve this problem, in the first part, an Anti-Coordination based POC Algorithm (ACPOCA) was proposed, which can efficiently reduce the iteration times of channel assignment process, and improve the network throughput. However, without a central controller, both the signaling and suspension time of the network are limited by the distributed setting. Therefore, to address such challenges, in the second part, a deep learning based, intelligent POC assignment algorithm with the centralized SDN is proposed. The contributions of the deep learning based proposal can be explained in two aspects.

- First, with the central control paradigm of SDN, switches do not need to exchange their channel states anymore. All channel assignment processes can be carried out in the central controller. Thus, the signaling overhead of the network is significantly reduced.

- Second, since the deep learning approach can learn from previous channel assignment processes through training with the data collected from the existing channel assignment algorithms (e.g., ACPOCA), the channel assignment can be finished in just single iteration.

In summary, this approach, which we refer to as the Deep Learning based Partially Overlapping Channel Assignment (DLPOCA), can efficiently reduce the suspension time caused by channel assignment, and achieves almost non-suspending flows during the channel assignment process.

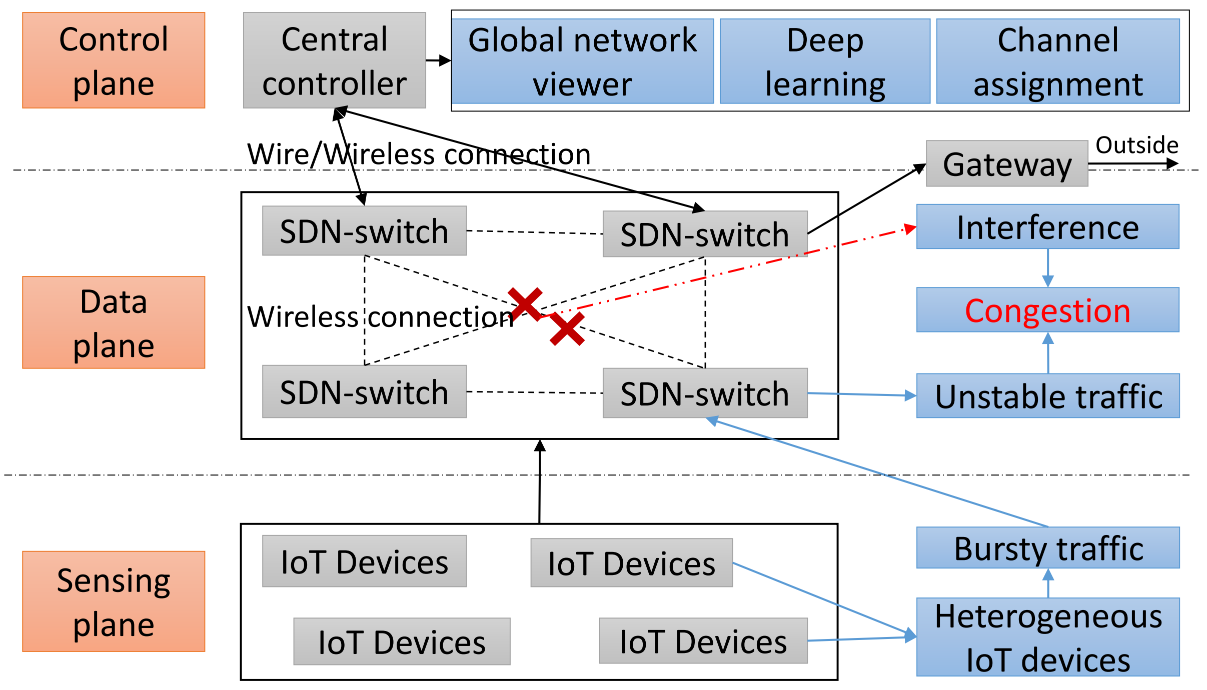


Figure 1.2: The SDN-IoT architecture.

Additionally, in existing channel assignment algorithms, as the most important baseline metric in the channel assignment, the traffic load is usually assumed to be continuous and stable. This means that the traffic load in the next time interval after the channel assignment is similar to that in last time interval. However, the real traffic loads in practical networks are more complex and may suddenly change like a bursty traffic. Particularly in SDN-IoT, in the sensing plane of the SDN-IoT structure, the devices can be divided into three groups depending on the sensing mechanism: periodic sensing, event-driven sensing, and query-based sensing [39, 40, 41, 42]. For the periodic sensing devices, such as temperature, humidity and light sensing devices, they sense data and periodically integrate and transmit them to the central controller. Moreover, these devices may have different policies (e.g., sensing circle, volume of sensing data, and so forth). For example, a kind of temperature sensing device may collect 3kB temperature once in every 30s, another kind of humidity sensing device may collect 10kB humidity data once every 1 minute. Those different policies result in highly complex, periodically bursty distribution of the traffic load. For the event-driven and query-based sensing devices, the traffic load is also not consequent but explosively generated when a new event occurs or a query comes. The bursty traffic caused by the event-driven and query-based sensing device is more random and irregular than that generated by the periodic sensors. In the SDN-IoT network, with heterogeneous resources, sensing devices can hardly cooperate with one another, making the switch impossible to know the real future traffic integrated by the heterogeneous sensors. Furthermore, besides the traffic generated by its connected sensing devices represented as integrated traffic, each switch may also have to forward the traffic from other switches denoted as relayed traffic. In practical networks, the mixed traffic containing integrated and relayed traffic becomes more complex. Even though many existing researchers proposed some methods about traffic load prediction, most of them focused on the traffic changes in the long-term and did not consider the traffic load change caused by routing. Therefore, in the third part of our research, a deep learning based prediction and POCs assignment algorithm is proposed. contributions of the third part are separately outlined as follows.

- First, I use the powerful deep learning approach to predict the complex traffic, which can achieve above 90 percent accuracy and have a quick response time. (5ms<)
- Second, I investigate the advantage of using the centralized SDN technique in the deep learning based traffic load prediction in the IoT environment. In order to show the improvement of deep learning based traffic load prediction in SDN-IoT compare with conventional IoT, we respectively design three traffic load prediction algorithms to suit three different control systems (i.e., centralized SDN control system, semi-centralized control system and distributed control system). After designing those three different prediction methods, we further compare the prediction accuracy in those three different control systems. The result shows that, the prediction accuracy of centralized SDN based prediction is always better than those in the two other systems.
- Finally, with the centralized SDN control, we combine the deep learning based traffic prediction and partially overlapping channel assignment, that uses the predicted traffic load as the criterion to perform the intelligent channel assignment. Such proposed intelligent partially overlapping channel assignment, which we refer to as TP-DLPOCA, can efficiently increase the channel assignment accuracy and processing speed of channel assignment. The simulation results demonstrate that both the throughput and delay in the SDN-IoT with our proposal are better than those of the conventional algorithms.

1.2.3 The deep learning based network traffic allocation in SDN-IoT

After the radio channels are intelligently assigned to each links in SDN-IoT. The potential routing paths from source node to destinations can be easily got. However, as we mentioned above, with the traffic increase, the nodes in the certain routing path may suffer extremely high burden. Then, how can we allocate the traffic flow to suitable path to

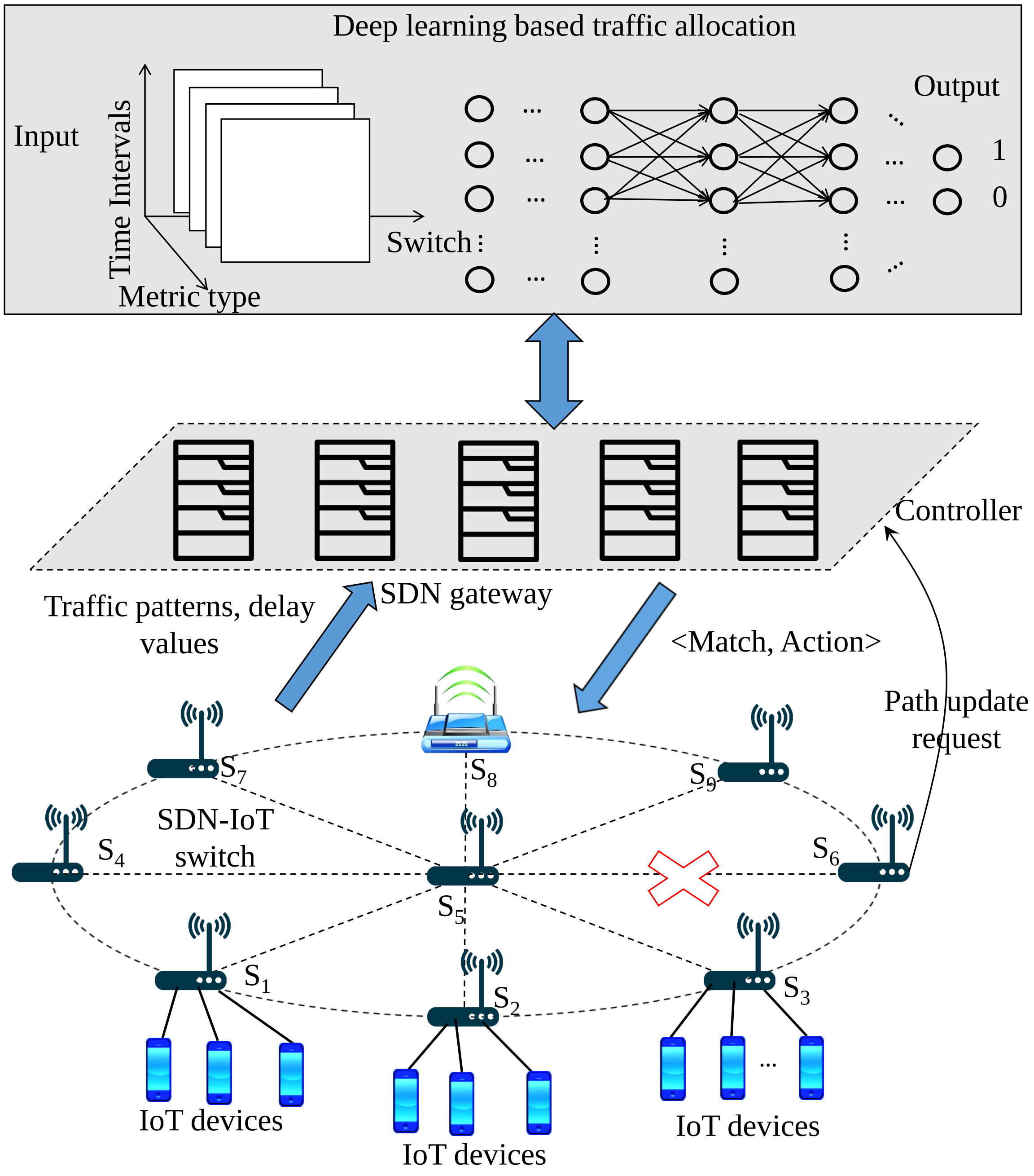


Figure 1.3: The considered SDN-IoT and deep learning based traffic allocation approach.

alleviate the network traffic burden? Current SDN enabled network still use conventional routing strategies which are commonly based on fixed rules [43], such as the Shortest Path (SP) algorithm. The problem of fixed rule-based routing protocols is that the same paths will be chosen when similar traffic patterns appear, even though these paths can result in traffic congestion according to previous experiences. Repetitions of the same fault lead to the unnecessary network performance deterioration. Moreover, the reactive manner utilized in conventional routing protocols, to update path after some link/switch failure, causes the unavoidable delay in a centralized control system [44]. Even we can develop an intelligent routing strategy by adding the memory mechanism, many other latent problems still exist and new problems may appear due to the changing network situations. Therefore, it is extremely important to design an intelligent routing strategy which has the ability to learn from previous experiences and adapt itself to the changes.

To address the aforementioned issues and considering the high computation resource equipped in the SDN-IoT controller, we propose a deep learning based intelligent routing strategy for SDN-IoT demenstrated in Fig.1.3. Definitely speaking, we consider the network as an image and the different features of traffic patterns as different channels of pixels. Since the Convolutional Neural Networks (CNNs) are the most widely utilized architectures in the field of image classification, in this article, I utilize CNNs to analyze the network traffic patterns and make the routing decisions. Our proposal consists of two phases, the initial phase and the running phase. The contributions of our proposal can be summarized as follows.

- I propose online self-learning method in network. This method will continuously label real-time collected data to retrain the deep learning architectures. Therefore, the deep learning architectures can get adapted to the environmental changes.

- To overcome the repeated mistakes caused by the fixed rule based routing algorithm, I utilize the deep learning method to predict the traffic state in SDN-IoT. Definitely speaking, in the actual running phase, the central controller will monitor the network performance and utilize the real-time performance as the feedback of the routing decision to periodically retrain the CNNs. Thus, the proposed strategy can not only learn from previous experiences and proactively update the paths, but also adjust and improve itself to suit the new traffic situation.

Chapter 2

Overview of Resource allocation, game theory and deep learning in IoT

2.1 Introduction

In IoT, the cellular communication and D2D communication are widely deployed in IoT as the underling technique to fast construct the feasible IoT. However, the related research works did not take into consideration the shortcoming of cellular infrastructure in remote, rural, or disaster-affected areas.

UAV based wireless networks recently emerged as an attractive technique for facilitating public safety and military communications [45]. This is because the UAVs can be rapidly deployed as aerial base stations to form a flexible cellular network [19, 46]. In [20], the deployment of a UAV-based communication network over a given geographical area was analyzed. The analysis demonstrated the feasibility of deploying UAVs in D2D enabled cellular networks. While the UAVs equipped with reasonably large storage and computing ability can be considered as content-centric server nodes in the considered IoT network [47, 48], On the other hand, one of our earlier works in [45] demonstrated how emerging wireless networks aided by UAVs can become useful in enabling communications in ultra-dense environments in urban locations which might be the hot area in future IoT scenario. to the best of our knowledge, no previous work has investigated the importance of a combined UAV and D2D based network technology for supporting the network in IoT.

However, with the mixed using of UAV, D2D and UAV, the heterogeneous devices and underling protocol deployed in distributed manner may cause unpredictable error and conflict. In order to solve such problem in the complex IoT environment, The SDN-IoT structure processed in centralized manner was first proposed by Qin at al. in 2014 [35] which incorporates and supports commands in a heterogeneous structure to optimize the SDN-IoT network. After the first SDN-IoT structure was proposed, many related works emerged. Sood et al [49] employed a multi-objective constraint to manage the layer resource in SDN-IoT. Ojo et al [13] presented a SDN-IoT architecture with Network Function Virtualization (NFV) implementation to address the new challenges of IoT. Nguyen et al [50] proposed a SDN-based IoT Mobile Edge Cloud Architecture to deploy diverse IoT services at the mobile edge.

With such SDN-IoT, as shown in Fig.1.2 the algorithm for network control and resource allocation methods can be easily deployed in SDN central controller to manage the whole IoT. About resource allocation, there are many exist works detailed introduced in the next sections. In The remain sections, we first overview the existing studies about resource allocation in conventional networks in Sec.2.2. Then, in Sec.2.3, the related radio allocation namely channel assignment algorithms are introduced. In this section, we first survey the channel assignment in wireless network and then introduce the usage of new partially overlapping channels (POCs) in wireless network and IoT. After the channel assignment, we simply introduced the research state about my employed techniques, game theory in network in Sec.2.4 and deep learning in network in Sec.2.5. Finally, the existing works about network traffic allocation are discussed in Sec.2.6

2.2 Overview of resource allocation in network

The network resource such as radio resources, computing resources, power resources are limited in the IoT. There are many resource allocation related researches, in this section, we introduce the research flow of resource allocation.

Ten years ago, the resource allocation algorithm is first proposed to deal with the limited resource utilization in cellular network. S.A. Grandhi et al., [51], gives resource allocation solution for cellular radio systems. In the research, a Distributed Dynamic Resource Allocation (DDRA) scheme based on local signal and interference measurements is proposed for multiuser radio networks. In [52] C.Curescu et al. presents a bandwidth allocation and admission control mechanism to be used in a radio network cell of a future generation telecommunication network. This approach is based on the time-aware utility to maximize the quality level of network. In the cellular network, the previous resource allocation algorithms are mostly just consider the user in the network directly connect to the base station, to improve it. M.Dohler et al. [53] proposed a FDMA-based regenerative multihop links based resource allocation method, and researchers in [54] proposed a relay and centralization based resource allocation algorithm. However, the resource limitation of the relay node are not fully considered in those researches. To address this issue, Y.

Li [55] proposed a relay station based integrated radio resource allocation algorithm which also take the resource limitation of relay station into consideration.

The resource allocation problem is widely studied in conventional cellular network, however, with the network technique developing, more complex heterogeneous network such as D2D and UAV communication enabled IoT have emerged. To conquer the challenge, many researches are proposed. To model the complex network environment, R. Yin [56] and F. Wang [57] use game theory to model the resource allocation problem, the power and radio resource allocation algorithm is proposed correspondingly. D.H.Lee [58] considers the resource allocation solutions in both distributed and centralized control manner in the network. In [59], the authors jointly consider admission, power and radio resource allocation, and propose a tress stage resource allocation algorithm. Then, B. Zhou in [60] and M. Hasan in [61] consider the relay situation in the D2D enabled cellular network. Those above methods mainly consider single factors in the network, the researches [62], [63] and [64] further consider power consumption in both user and station, the solutions are suitable for heterogeneous networks. However, those methods lack consideration of multiple cellular and the interferences between different cellulars, Z. Zhou in [65] considers C-RAN and multiple cellular environment, proposes an energy-efficient resource allocation algorithm for D2D Communications network. Those resource allocation methods widely research the balance of different types of resource in the network. However, all of those researches lack the focus on specific radio resource features, and there are no related resource allocation algorithm designed for SDN-IoT.

2.3 Overview of channel assignment in wireless network

In this section, we give a preliminary of the specific channel resource allocation in wireless network. The radio channel allocation (i.e., assignment) is widely researched in conventional wireless network.

2.3.1 Channel assignment in wireless network

A. Raniwala et al. in [66] at first studies the channel assignment problem in multi-channel wireless mesh network and proposes that the channel assignment problem can be treated as graph coloring problem [67] which is a NP-hard problem [68]. P. Kyasanur in [69] proposes a classical fixed channel assignment algorithm in the multi-chennal multi-interface wireless network. This method uses stable channels to fit for static network, however, when the number of channels increases, the proposed NICs based interface model may not satisfy the limited resources. In [70], instead of fixed network topology, the authors propose a topology channel based channel assignment algorithm. In this approach, the channel assignment scheme is based on the independent network traffic to enable flexible network topology. Even through the method is designed for flexible network, the channel assignment algorithm itself is not dynamic. In this vein, some dynamic channel assignment algorithms are proposed. The study in [71] shows a technique to allow the channel dynamic switch within significant shot time slot. With such kind of quick channel switch technique, A Raniwala et al. in [72] proposes traffic load based dynamic channel assignment algorithm. A spanning tree topology is used in this algorithm to leverage routing overload in the considered wireless network, both the neighbor-to-interface binding and interface-to-channel binding are considered in this approach. In [73], the authors further consider the impact of traffic patterns and network connectivity of wireless links in network, a corresponding fixed channel assignment algorithm is proposed. The priority rank of all nodes in network is calculated as the main factor of this proposal. And the author first considers the nodes near the gateway with highest traffic load are allocated highest ranks in the algorithm. In addition, to consider more complex network environment, Zhou et al. [74] studies the radio allocation problem in multi-channel multi-radio network, in this research, the authors focus on the minimal video distortion and resource fairness to improve channel utilization. All of those studies work on the non-overlapping (orthogonal) channels, however, in many cases, the application of partially overlapping channels gives much better performance in terms of both network throughput and QoS. In next part, I give a overview of partially overlapping channels in wireless network.

2.3.2 Partially overlapping channel assignment in wireless network

in 2005, Mishra et.al [75, 30] first research the situation of using partially overlapping channels (POCs) in the network can improve the network performance. Bukkapatanam et al. [31] then gives a detailed analysis of usage of overlapping channels in backbone network. However, no corresponding POCs based assignment algorithm is proposed.

As the channel assignment algorithm is proved to be a NP-hard problem, In [76], a heuristic POCs assignment algorithm is proposed. However, in this approach, the network traffic is static and the traffic load in each node are not considered. P.D.F et.al [77] firstly consider the POCs assignment problem from the game theoretical perspective. With the definition of utility function of nodes (players) in network, the players auction is processed to make the total utility of network maximum. After it, the same authors further improve the game theory based POC assignment algorithm with cooperative game theory[29] and proved the existence of the steady state(Nash Equilibrium) in the POC assignment algorithm (game). Besides, without employ game theory, there are some other POCs assignment studies, Y. Ding in [78] proposes a POC assignment algorithm based on unknown network traffic. Instead of conventional wireless network, P. Ciotrnae et.al [79] considers the POC problem in WI-FI communication, with the build test bed, both the interference model and SINR performance are detailed analysis. F. S. Bokhari [80] proposes a mixed POCs assignment algorithm in both centralized and distributed control manner. In summary, the related works widely study the POCs assignment in wireless networks, many heuristic algorithms are proposed to minimize the interference and improve the network throughput. However, as I know, no POCs assignment algorithm is proposed for the specific IoT environment. In next part, we investigate the channel assignment researches in IoT.

2.3.3 Channel assignment in IoT

The IoT is emerged as the famous network around the world recently, in terms of the new features and special heterogeneous structures in IoT, many novel channel assignment algorithms for IoT are proposed.

As the D2D is the main technique used in IoT, many researches work on the channel assignment in D2D Underlaying cellular network that I have introduced in section.2.3.1. Recently, in [81], Thong Huynh et.al further considered the joint downlink and uplink interfere problem in D2D Underlaying cellar network. N. ul Hasan [82] investigates the IoT scenario in 5G network. Based on the specific architecture and infrastructure, a QoS-aware channel assignment mechanism for smart building with heterogeneous IoT devices is proposed. Consider the channel assignment and network traffic allocation problem are not independent from each other. HyungWon Kim in [83] proposes a mixed channel assignment and routing algorithm which are suitable for event-driven video traffic in wireless IoT. Last year, in the study of [84], H. B. Salameh considers the cognitive radio networks (CRNs) in IoT, a time sensitive channel assignment approach under proactive jamming attacks is proposed to improve the network performance. In this approach, the security features of licensed users activities, fading conditions, and jamming attacks are jointly considered. The IoT relies on cellular network, and in order to connect to various IoT devices, the D2D connections are cooperated with cellular IoT. L. Zhao consideres the the interference graph in the cooperated D2D Cellular Networks in IoT and proposes a greedy based channel assignment algorithm. Those researches widely investigate channel assignment in IoT, however, none of them consider the partially overlapping channel can be used in IoT to improve the radio resource utility and network performance. In my thesis, the POC assignment in IoT by using game theory and deep learning is the main study objective.

2.4 Overview of game theory in network

As mentioned above, the channel assignment problem is a NP hard problem. Game theory, as a famous tool to provide near optimal solution to deal with NP hard problem, is widely used in economics area [85, 86]. In last decade, a lot of engineering issues are solved by researchers using game theoretical perspective. In the specific network area, the game theory based solutions are widely distributed in topics such as network traffic allocation [87, 88, 89], power resource allocation [90] and CRAN deployment [91]. Meshkati in [92] use non-cooperative game theory based algorithm to allocate power, traffic in network. In this research, the author consider the QoS level of user and let users to play the non-cooperative game to achieve the maximum utility in terms of energy and transmission delay. However, the channel assignment is not considered in this study, therefore, in [93], D. Niyato considers the radio allocation problem in network scenario which using IEEE 802.16-based multi-hop wireless mesh infrastructure for relaying traffic from IEEE 802.11 Wireless Local Area Networks (WLANs) based communications. In this study, both the bandwidth and admission are allocated by using game theory based auction process. In [94], Z. Zhao et.al use incompletely cooperative game theory to solve the channel assignment and system performance optimization problem in wireless mesh network.

The above methods use game theory to solve problems in network, however, none of them consider POC assignment. In [95] Y. Song considers the jointly power and channel resource allocation in access network. Furthermore, in [96], W. Yuan proposes a overlapping channel capacity optimization algorithm in WLAN based on game theory.

In [29], P.D.F et.al employ game theory concepts to model mesh routers as decision makers of a cooperative game. In the cooperative game, the interaction among all mesh routers can be classified as an identical interest game. In this proposal, a players negotiation based POC assignment algorithm is proved that can converge to a steady state (Nash Equilibrium). However, in the above approaches, they do not consider the effect of algorithm convergence time and how to adjust the algorithm to the highly dynamic network scenario. Furthermore, in the existing works, many channel assignment algorithms are based on the traffic loads of nodes without taking into account the situation of dynamic traffic load. In the dynamic traffic load scenario, the traffic load of each load may change frequently which means that the channel assignment should be correspondingly changed also. In addition, the dynamic topology of the network was not considered by existing research works whereby the nodes may move frequently, which is the common case in IoT. In such a case, the distance between each node may dynamically change and the condition of the respective links change correspondingly. In other words, when the network topology changes, the channels should also be reassigned to cope with the new topology. However, the long convergence time and need of global information make it difficult to reassign the channels frequently. Because both the dynamic traffic load and dynamic network topology exist in our considered combined UAV and D2D network, the existing works may not be applicable to such networks. Therefore, in my thesis, a new anti-coordination game based POC assignment algorithm in section. 3 and the further improved deep learning based POC assignment algorithm in section. 5 are proposed.

2.5 Overview of deep learning in network

The performance of deep learning has been significantly improved since Hinton *et al.* [97] proposed the greedy layer-wise training method to pre-train the deep belief architectures. As more layers in the structures can represent a more complex relationship between the input and output, deep learning has become an efficient tool to explore the unknown relationships among a number of factors. Besides the academic research on its applications in image classification and nature language processing [98], various enterprises have adopted deep learning to promote their products and improve their services. For example, Apple's "Siri" utilized this technique to provide the best response to customers' requests [99]. In the field of communication networks, researchers also attempt to adopt this technique to address the emerging network challenges. However, not many achievements have been made due to the difficulty in characterization of the deep learning structure's input and output for defining networking problems [100, 101]. Wang *et al.* applied the deep learning technique to find the features of the traffic flow data [102]. The results showed that their approach works well for protocol identification and anomalous protocol detection. In [101], He *et al.* present an efficient green resource allocation algorithm based on the deep reinforcement learning, which can achieve high Quality of Experience (QoE) performance. To meet the fast convergence requirement of the future backbone network, our earlier work [103] proposed a tensor based deep learning approach to solving the routing problem. We considered utilizing a tensor to arrange the multiple parameters related to routing performance and the simulation results evaluate the efficiency of the deep learning strategy. deep learning based routing strategy running in GPU accelerated Software Defined Routers (SDRs) which can be widely deployed in SDN-IoT. That work demonstrated that the accuracy of the deep learning structure reaches as high as 95% and the GPU accelerated routers conducts the computation 100 times faster than the conventional routers. However, the performance of the deep learning structures in [102, 103] depends on the supervised training which needs a large quantity of data. However, all of above works not consider the channel assignment problem in wireless network especially in IoT. In my thesis, we consider the deep learning for both radio channel allocation and network traffic allocation. In the proposals, the network traffic patterns and corresponding decision data are used to train the proposed DBN and CNNs for the decision making of both the channel assignment and traffic routing. While the controller utilizes the deep neural networks to choose the channel and paths, the historical experiences are continuously collected and formatted as new training data.

2.6 Overview of deep learning in network traffic allocation

Our earlier work in [100] envisioned the first proof-of-concept of using deep learning architectures for substantially improving the heterogeneous network traffic control. A deep learning system was proposed that can be trained in a supervised manner based on uniquely characterized inputs using traffic patterns at the edge routers of a wireless backbone network. However, the deep learning algorithm was trained upon a considered benchmark routing method, namely Open Shortest Path First (OSPF). The survey conducted in [104] demonstrated that there exist different deep learning architectures such as Deep Boltzmann Machines (DBMs), Deep CNNs, and so forth that could be exploited for network traffic control systems. However, the case study considered in that work also considered a baseline routing method for training the deep learning algorithm. Furthermore, the work in [103] explored current Software Defined Router (SDR) architectures and demonstrated how the deep learning technique can be harnessed to compute the routing paths. The Graphics Processing Unit (GPU)-accelerated SDR enabling massively parallel computing for the deep learning was shown to substantially improve the backbone network traffic control. However, similar to the afore-mentioned researches, this work also adopted a supervised deep learning system dependent on a conventional rule-based routing method. Therefore, how to design a new traffic allocation algorithm which is not depended on the existing labeled data and get rid of conventional routing protocol in SDN-IoT becomes a new challenge.

2.7 Summary

In this chapter, I first give a preliminary on the related works of resource allocation in conventional wireless network and the considered SDN-IoT. From the existing works, there are lack of methods to deal with the partially overlapping channels in IoT, and the conventional channel allocation methods only focus on the static scenario and not intelligent for dealing with the changing traffic and topology in the IoT. To address such issues, I introduce two powerful tools namely game theory and deep learning and survey the related applications of using the tools in network. Besides, I investigated the research flow of deep learning used in network traffic resource allocation area. The proposed novel deep learning based traffic allocation algorithm is detailed described in chapter.6. With the overview of existing works, the weakness of past methods and the challenges of the resources allocation problem in IoT are emerged. In next chapter, we detailed model the considered system and formulate the radio resource allocation problems. The solutions for solving the problems are correspondingly proposed individually in next chapters.

Chapter 3

The Proposed Anti-Coordination Game based Channel Assignment

In this chapter, we detailed describe how to model the channel assignment problem in considered IoT and propose the corresponding anti-coordination game based channel assignment algorithm.

To make the proposal clear, in this part, I firstly give the network model and channel interference model. By analyzing the models, the partially overlapping channel assignment problem is formulated as a game theory based utility maximization problem. Consider the complexity of the UAV-enabled IoT, we proposed the anti-coordination game based channel assignment algorithm referred as ACPOCA to solve the utility maximization problem in two steps. In the first step, we considered the topology of the network is fixed, the corresponding POC assignment solution are proposed in section. 3.2.2. Then, the full ACPOCA to handle the dynamic topology is proposed in section. 3.2.3. In section. 3.3, I clarify the existence and uniqueness of the steady state of our proposed algorithm. The simulation results are given and analyzed in section. 3.4.2.

3.1 Network Model

Consider the UAV-enabled IoT network as a three-dimensional topology. Also, consider the UAV and IoT Devices sharing the same channels. Therefore, in the remainder of the paper, we refer to both the UAVs and IoT devices as "nodes". Let N denote the number of existing nodes in the system that are represented by the set, $\mathbf{A_{old}} = \{a_1, a_2, \ldots, a_N\}$. Each node in the considered network is represented by its own features, i.e., latitude, longitude, and height. In contrast to the traditional wireless network, the considered IoT exhibits high flexibility, and nodes can move and be added or removed according to situational demands. If M nodes are added to the network, they are denoted by

$\mathbf{B} = \{b_1, b_2, \ldots, b_M\}$. Therefore, the total nodes, in the considered system, are represented by $\mathbf{A} = \mathbf{A_{old}} \cup \mathbf{B} = \{a_1, a_2, \ldots, a_N, a_{N+1}, a_{N+2}, \ldots, a_{N+M}\}$.

In order to transmit the IoT data, the nodes in our considered network are assumed to comprise 802.11 2.4 GHz links and multiple radios with up to 11 channels. As mentioned earlier, both primary cellular and D2D links share those channels. However, the non-overlapping channels are limited (e.g., channels 1, 6, and 11). On the other hand, using overlapping channels in an arbitrary fashion results in severe interference and eventually network congestion. In order to alleviate this problem and improve the aggregate throughput of the considered network, we aim to exploit POCs. Even though POCs can also interfere with each other, their interference range is significantly smaller than the typical overlapping channels [105]. Such reduced interference range of POCs enables an increased number of parallel transmissions, and, thus, leads to increased network capacity.

The issue of assigning POCs can be considered to be an optimization problem in which the available communication channels need to be mapped to network interfaces for minimizing signal interference and maximizing the communication capacity. The interference range is defined as the distance within which interference occurs. Furthermore, in a network having multi-channels connections, there are four different types of interferences which should be addressed due to their influence of network capacity: co-channel interference, orthogonal channels interference, adjacent channels interference, and self interference [29]. Next, we present a model to describe these different types of interferences.

3.2 Interference model

The *Interference Matrix* or "*I-Matrix*" method in [105] may be used to model the above-mentioned types of interferences in order to carry out appropriate channel assignment. *I-Matrix* employs a special matrix to record the interference of each node and determines whether the chosen channel is viable or not to a given link exploiting POC. In order to record the interference of each node, a metric called *Interference Factor* (*IF*) is defined to measure the interference between channels. *IF* represents a ratio of geographical distance and interference range between two operating radios. $f_{p,q}$ expresses the effective spectral overlapping level between channels p and q.

The works in [106, 107] conducted experiments to measure $f_{p,q}$ under real conditions for different channel separations. Here, we use the result of those works to construct Table 3.1, Here δ refers to the interference range for a channel separation between channels p and q, $IR(\delta) = |p - q|$ denotes the geographical interference distance between channels p and q.

Now, let d refer to the distance between nodes operating with channels p and q. If the nodes use the same channel, d is set to zero. Then, $f_{p,q}$ is calculated in the following three cases respectively:

Table 3.1: Interference Range (IR).

δ	0	1	2	3	4	5
$IR(\delta)$	132.6	90.8	75.9	46.9	32.1	0

1. $f_{p,q} = 0$: when $\delta \geq 5$ or $d > IR(\delta)$.

   When the nodes are assigned orthogonal channels or have enough distance to avoid interference, no interference occurs between the radios.

2. $1 < f_{p,q} < \infty$: when $0 \leq \delta < 5$ and $d \leq IR(\delta)$.

   When overlapping interference occurs, the distance between the nodes is smaller than the interference range. In this case, *IF* should be a ratio proportional to the distance between the nodes. *IF* can be calculated as follows:

$$f_{p,q} = IR(\delta)/d. \tag{3.1}$$

3. $f_{p,q} = \infty$: when $0 \leq \delta < 5$ and $d = 0$.

   This happens because of the self interference problem. Hence, two overlapping channels ($\delta < 5$) are not viable to be assigned to the node due to their full interference.

After we have modeled the interference factor of POC, we further use the *Interference Vector* and *I-Matrix* to measure the interference situation of each node.

Table 3.2: *Interference Vector.*

d_i	*Ch1*	*Ch2*	*Ch3*	*Ch4*	*Ch5*	*Ch6*	*Ch7*	*Ch8*	*Ch9*	*Ch10*	*Ch11*
d_3	$f_{3,1}$	$f_{3,2}$	$f_{3,3}$	$f_{3,4}$	$f_{3,5}$	$f_{3,6}$	$f_{3,7}$	0	0	0	0

- *Interference Vector:* The *Interference Vector* is shown in Table 3.2 that is calculated based on all the *IF*s between one channel to all 11 channels. The table keeps track of the distance d_p to the nearest assigned radio in channel p.

- *I-Matrix:* Each node updates its own *Interference Vectors* of all 11 channels, which form an *I-Matrix* according to Table 3.3. Also, the node updates *I-matrix* when any channel assignment is changed.

Based on the afore-mentioned network and interference models, we are now ready to formulate the research problem.

Table 3.3: *I-Matrix.*

Ch	d_i	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*
1	d_1	$f_{1,1}$	$f_{1,2}$	$f_{1,3}$	$f_{1,4}$	$f_{1,5}$	0	0	0	0	0	0
2	d_2	$f_{2,1}$	$f_{2,2}$	$f_{2,3}$	$f_{2,4}$	$f_{2,5}$	$f_{2,6}$	0	0	0	0	0
3	d_3	$f_{3,1}$	$f_{3,2}$	$f_{3,3}$	$f_{3,4}$	$f_{3,5}$	$f_{3,6}$	$f_{3,7}$	0	0	0	0
4	d_4	$f_{4,1}$	$f_{4,2}$	$f_{4,3}$	$f_{4,4}$	$f_{4,5}$	$f_{4,6}$	$f_{4,7}$	$f_{4,8}$	0	0	0
5	d_5	$f_{5,1}$	$f_{5,2}$	$f_{5,3}$	$f_{5,4}$	$f_{5,5}$	$f_{5,6}$	$f_{5,7}$	$f_{5,8}$	$f_{5,9}$	0	0
6	d_6	0	$f_{6,2}$	$f_{6,3}$	$f_{6,4}$	$f_{6,5}$	$f_{6,6}$	$f_{6,7}$	$f_{6,8}$	$f_{6,9}$	$f_{6,10}$	0
7	d_7	0	0	$f_{7,3}$	$f_{7,4}$	$f_{7,5}$	$f_{7,6}$	$f_{7,7}$	$f_{7,8}$	$f_{7,9}$	$f_{7,10}$	$f_{7,11}$
8	d_8	0	0	0	$f_{8,4}$	$f_{8,5}$	$f_{8,6}$	$f_{8,7}$	$f_{8,8}$	$f_{8,9}$	$f_{8,10}$	$f_{8,11}$
9	d_9	0	0	0	0	$f_{9,5}$	$f_{9,6}$	$f_{9,7}$	$f_{9,8}$	$f_{9,9}$	$f_{9,10}$	$f_{9,11}$
10	d_{10}	0	0	0	0	0	$f_{10,6}$	$f_{10,7}$	$f_{10,8}$	$f_{10,9}$	$f_{10,10}$	$f_{10,11}$
11	d_{11}	0	0	0	0	0	0	$f_{11,7}$	$f_{11,8}$	$f_{11,9}$	$f_{11,10}$	$f_{11,11}$

3.2.1 Problem Formulation

Each node in the proposed network shown in Fig. 1.1 wants to be assigned a proper channel to maximize its throughput based on its own traffic demands. However, each node also wants its channel to be different from its neighboring node such that the interference is minimum. When the nodes improve their own connectivity through assignment of proper channels, the total connectivity will be improved also. However, this means that each node acts selfishly to obtain the best possible channel assignment. Without the help of a central controller (e.g., a ground station), the nodes need to use a distributed channel assignment procedure. In such a distributed scenario, the channel assignment problem can be represented by the properties of an anti-coordination game played by the nodes.

- Anti-Coordination Channel Assignment Game: Games such as the game of chicken and hawk-dove game in which players score the highest when they choose opposite strategies are called anti-coordination games. We use the Anti-Coordination game property that if and only if the strategy in the game has a total bandwagon, it satisfies the interference property of the channel assignment model. In our network, each node is considered as a *decision maker* of the game, and the assignment of channel is considered as a strategy. Thus, we can model the interactions among nodes as an anti-coordination channel assignment game. The game has finite sets of nodes, referred to as players $\mathbf{A} = \{a_1, a_2, \ldots, a_N\}$ with a common strategy space $\mathbf{S}$. In our work, we assign the channel(s) to a node's (i.e., player's) radios by its chosen strategy. We express the strategy of the i^{th} player as $\mathbf{s_i} \in \mathbf{S}, \mathbf{s_i} = \{k_{i,1}, \ldots, k_{i,c}, \ldots, k_{i,|C|}\}$, where $k_{i,c}$ is a binary value. When channel c is assigned to a player, we set $k_{i,c}$ to 1, and 0 otherwise. $|C|$ refers to the number of channels for the channel set $\mathbf{C}$. The Cartesian product of the players' strategy vector is defined as the game profile of the network, $\mathbf{\Psi} = \times_{i \in \mathbf{A}} \mathbf{s_i} = \mathbf{s_1} \times \mathbf{s_2} \times \cdots \times \mathbf{s_N}$. A game profile is composed of each strategy of every player. $\mathbf{s_{-i}}$ means the strategy set chosen by all other players except player i.

- Player Utility: The objective of the game is to maximize the network throughput. However, in the anti-coordination game, a player only focuses on his utility. We define the utility of a player i as M_i. This utility can be a proportional measure of the connectivity of each node as shown in (3.2). Each link with channel q's capacity is evaluated according to its interference factor, denoted by IF_q. The link data rate R is used to measure the traffic load of the player (node). The importance of a player also depends on two topology control factors, h and k, which mean its hop count to the gateway (GW), and whether it can connect to the GW or not. Here, h and k are used to measure how efficiently these links connect to the gateway (GW). The work in [29] assumes that the utility is linearly proportional to the hop count h. However, that assumption has a shortcoming when the network is large whereby the utility of any node far away from the gateway decreases quite fast and finally approaches 0, and therefore, is eventually ignored in the next anti-coordination game. Hence, in this work, we adopt a natural logarithmic function $\ln(h+2)$, where $h+2$ is used to avoid the denominator of the utility function to become 0 and, this exhibits better performance in larger networks. k is set to 1 if the node can indirectly reach the GW, and 0 otherwise.

$$M_i = k\frac{\sum_{q\in\mathbf{C}} \frac{R}{IF_q+1}}{\ln(h+2)} \tag{3.2}$$

- Social Welfare: The social welfare means the total utility of the network. Each player has its utility function $U_i(\mathbf{\Psi})$ dependent on its own strategy and other players' strategies. Because we defined an anti-coordination game, the social welfare of the game, U_{NET}, can be represented as follows.

$$U_{NET}(\mathbf{\Psi}) = U_i(\mathbf{\Psi}) = \sum_{i\in\mathbf{A}} M_i, \quad \forall i. \tag{3.3}$$

By modeling the channel assignment as an anti-coordination game, we may use the game theoretical properties to guarantee optimized network performance. In such a game, the players will change their interdependent strategies in $\mathbf{S}$ to improve their utilities, which correspondingly improve the value of U_{NET}. Then, several important issues arise: (*i*) how the players play the game to improve the social welfare, (*ii*) whether they ever reach a consensus, or steady state, (*iii*) if the topology changes, how the game goes on to reach such a steady state, and (*iv*) how efficient this steady state performance would be. In the following section, we propose an algorithm to allow the nodes to play such a game and address the afore-mentioned issues by proving the existence of a steady state and evaluating its performance.

In this section, our main objective is to design an optimal channel assignment algorithm using game theoretic approach. In this vein, we first show that our formulated game is a potential game [108].

In our prior research in [29], if a game is in a state of *Nash Equilibrium* (NE) whereby the players arrive at an agreement, the game can be considered to be in a steady state. *Strategy* $\mathbf{s}^* \in \mathbf{S}$ is an NE if the game utility satisfies the following,

$$U_i(\mathbf{s}^*) \geq U_i(\mathbf{s_i^{'}}, \mathbf{s_{-i}}) \ \forall \ \mathbf{s_i^{'}} \in \mathbf{S_i}, \forall i \in \mathbf{A}. \tag{3.4}$$

In our formulated game in Sec. 3.2.1, there exists a *potential function* P as follows,

$$P(\mathbf{s}^{'}, \mathbf{s_{-i}}) - P(\mathbf{s}^{''}, \mathbf{s_{-i}}) = U_i(\mathbf{s}^{'}, \mathbf{s_{-i}}) - U_i(\mathbf{s}^{''}, \mathbf{s_{-i}}) \ \forall \ i, \mathbf{s}^{'}, \mathbf{s}^{''}, \tag{3.5}$$

where $\mathbf{s}^{'}$ and $\mathbf{s}^{''}$ stand for two arbitrary strategies. It is straightforward that the network utility function (3.3) itself is a potential function for the game. Hence, we have,

$$P = U_i(\mathbf{\Psi}) = U_{NET}(\mathbf{\Psi}), \forall i. \tag{3.6}$$

Thus, our considered problem is a potential game. In *potential games*, the existence of NE can be proved. Also, such games have several useful properties. The first property is that the finite potential game possesses at least one pure strategy NE [108]. The second property is that All NEs are either local or global maximizers of the utility function [108]. The third property states that there are well-known learning schemes to reach these function maximizers such as *best response* and *better response* [109].

By these properties of a potential game, we can prove that our formulated game can reach a steady state, and all players will reach a consensus. Now, let us call a player a_i unhappy, if a_i can achieve better utility by changing its channel. Let $\mathbf{A_u}$ indicate the set of unhappy players. We now run the learning schemes to make the unhappy player happy until no unhappy node exists, i.e., ($\mathbf{A_u} = \emptyset$). With potential and coordination games, learning schemes like best response, better response, smoothed better response, and perfect foresight response may be used to accomplish such goals. These learning schemes are described below.

- **Best response:** As expressed in (3.7), the player searches its entire strategy space and selects the one which yields the best outcome considering the other players' strategies. This scheme provides fast convergence in polynomial time. In fact, in our game, the number of steps is equal to the number of connected links in network. On the other hand, it requires intensive processing that grows linearly according to the strategy space and has normal probability to get trapped in a local optimum.

$$rCl\mathbf{s_i^{t+1}} = \arg\max_{\mathbf{s} \in \mathbf{S_i}} M_i. \tag{3.7}$$

- **Better response:** As expressed in (3.8), each player selects a random strategy and keeps it as long as it generates a better outcome than the previous one. Thus, better response provides a less intensive computation at the cost of a slower convergence to the equilibrium, and has normal probability to be trapped in a local optimum.

$$rCl\mathbf{s_i^{t+1}} = \begin{cases} \mathbf{s_i^{\text{rand}}} & \text{if } M_i(\mathbf{s_i^{\text{rand}}}, \mathbf{s_{-i}}) > M_i(\mathbf{s_i^t}, \mathbf{s_{-i}}) \\ \mathbf{s_i^t} & \text{otherwise.} \end{cases} \tag{3.8}$$

- **Smoothed better response:** This method uses randomness in the decision process which may lead to convergence to the global NE with a high probability. This uncertainty occurs according to the following probability function:

$$p(\mathbf{s_i^{\text{rand}}}, \mathbf{s_i^t}) = \frac{e^{M_i(\mathbf{s_i^{\text{rand}}}, \mathbf{s_{-i}})/\gamma}}{e^{M_i(\mathbf{s_i^{\text{rand}}}, \mathbf{s_{-i}})/\gamma} + e^{M_i(\mathbf{s_i^t}, \mathbf{s_{-i}})/\gamma}}, \tag{3.9}$$

  where γ is a parameter responsible to control the trade-off between the technique's outcome performance and convergence speed. A large value of γ enables an extensive strategy search and slow convergence. On the other hand, a small γ restricts the search while improving the convergence speed. The player will evaluate the newly selected random strategy against the previous one, and select the new strategy according to (3.9). Thus, notice that smoothed better response incurs the least intensive computation at the cost of the slowest convergence to NE.

- **Perfect foresight response:** While this is similar to the best response technique, it involves the players to form an expectation by discounted average time of action distributions of the next period instant of current action distribution [110]. This method gives one path from any initial state to the NE, but it may be trapped in this point forever. Also, compared to the best response technique, this may result in a higher computation cost.

The above four learning schemes have their own advantages and disadvantages. If the nodes were involved in a cooperative game as shown in our earlier work in [29], smoothed better response technique could be used. However, in such a scenario, when each node changes its channel, all other nodes should update their *I-Matrix*. This means that the nodes changing their channels need to exchange their channel selection information with the other nodes, thereby significantly increasing the signaling overhead. In addition, with the cooperative game method, each node calculates its utility depending on the strategy selection of all other nodes and the global information. Thus, when the traffic load and topology of the network change, the utility of every node is changed and the channels need be reassigned to all the nodes. During the reassignment of channels, the content transmission and delivery of the whole network may be halted. Both the increase of signaling and waiting time to perform reassignment will cause severe degradation of throughput of network. Therefore, in order to avoid the global channel reassignment and decrease the signaling overhead, our algorithm should be designed independent of other nodes and to converge as fast as possible. In other words, different from the cooperative game in [29], in our anti-coordination game played by the nodes in a decentralized manner, each player's response only depends on its own utility from local information and does not need to know the utility of the other players. As a result, the channel reassignment only affect local node and neighbors and signaling overhead can be reduced. The performance of decreased overhead, P_{sig}, can be calculated as follows:

$$P_{sig} = 1 - \frac{|E|}{|E| \times N_{iter} \times |C|}, \tag{3.10}$$

where $|E|$ denotes the number of connected links in the network. N_{iter} stands for the number of iterations by smoothed-better-response.

Thus, we design our Anti-Coordination game based Partially Overlapping Channels Assignment (AC-POCA) algorithm using the best response technique to allow it to converge rapidly and also avoid global channel reassignment. The steps of AC-POCA are shown in Alg. 1, in which we assume each node has a unique identification parameter $ID_{a_i \in A}$ for routing purpose. It is worth reminding that due to the features of our considered UAV-enabled IoT network, the network has a highly dynamic topology and high mobility of nodes. We respectively describe our algorithm in two steps, first we consider the static topology of the network and then we consider the dynamic case.

3.2.2 Static topology

In the initial phase, all the nodes are initialized such that they belong to the unhappy set, $\mathbf{A_u}$. Each node uses a priority queue to store $\mathbf{A_u}$. The priority order of each node in set $\mathbf{A_u}$ is decided by various metrics such as traffic load, number of neighbors, distance to gateway, and so forth. Here, we use a queue to store the unhappy set because with the best response technique, each node only performs channel assignment once (i.e., only one iteration) and will not affect the channel selection of nodes carrying out channel assignment prior to it (i.e., in front of it in the queue). Then consider the I-matrix of each node also initial calculated in the first case, consider the de-centralized network, where each node only calculates the distance from other nodes in each interfere vector within its transmission range. Then, each node processes the steps is shown within lines 13 to 27 in Alg. 1 so that the best response strategy is used to assign the POCs. In line 19, after a channel is selected by the calculated utility, a valid threshold h_v, denoting the tolerance to channel interference is used to assess whether the channel selection strategy is valid or not. Then, the node removes itself from the unhappy set $\mathbf{A_u}$ and broadcasts $ID_{a_i \in A}$ and notification qt to other nodes to continue those steps until none of the nodes belongs to $\mathbf{A_u}$.

From here on, we present an easy-to-understand example of our proposed algorithm in Fig. 3.1 that describes a simple scenario where only 4 nodes construct a network. For the sake of simplicity, consider that the distance between each node is 100 meters and the link data rate is 1Mbps for all links. The assigned channels and overall network utilities for different initial orders of the nodes are listed in Table 3.4. The table demonstrates the different steady states of each link in different initial orders. For example, in row 1, the order {1,2,3,4} means the nodes a_1, a_2, a_3, and a_4 have the first, second, third, and fourth initial orders, respectively. According to their orders, the four nodes are placed into the unhappy set $\mathbf{A_u}$. Then, the first order node, i.e., a_1 is selected to play the channel assignment game. Node a_1 performs channel assignment on all of its links represented by edges $\{e_1,e_2\}$. e_1 chooses the first strategy s_i^{sel} from strategy space$\{k_{i,1}, \ldots, k_{i,c}, \ldots, k_{i,|C|}\}$, and judges whether it satisfies (3.7). Because e_1 is the first link which the first channel assignment is performed, no other channel will interfere with it. Thus, using line 13 of Alg. 1, e_1 chooses channel 1 and its interference factor, *IF*, is zero. In addition, using (3.2), the utility of a_1 for e_1 is 0.721. If a_1 changes to any other channel on e_1, its utility will not increase. If the utility is larger than a threshold (e.g., 0 in this example), a_1 considers the channel assignment on e_1 to be valid. Similarly, a_1 assigns channel on its remaining link/edge e_2. Because the channel assigned on e_1 interferes with that on e_2, a_1 needs to choose the best channel (e.g., channel 6) so as to make its utility for e_2 the maximum (i.e., 0.721). Thus, a_1's utility becomes 1.442. After all its edges have received valid channel assignments, a_1 becomes a happy player, and therefore, is removed from $\mathbf{A_u}$. At this point, a_1 broadcasts this event to the other nodes, which, in turn, update their *I-Matrix*. Now, a_2 becomes the first order node in $\mathbf{A_u}$ that starts to play the channel assignment game on its links e_1 and e_4. Because a channel was already assigned on e_1, a_2 only needs to assign a channel on e_4, which interferes with the channels assigned on e_1 and e_2. Using (3.2), a_2 decides that the best channel to be assigned on e_4 is 7, which yields a_2's utility to the maximum (i.e., 1.822). As a consequence, a_2 is removed from $\mathbf{A_u}$, and a_3 starts playing the channel assignment game on only link e_3, which is within the interference range of the other three edges (e_1, e_2, and e_4). Regardless of the chosen channel, a_3's *IF* for e_3 is ∞ mentioned in Sec.3.2 leading to a utility of 0, which is invalid. Therefore, e_3 is not assigned any channel. For agreeing with the channel already assigned on e_2, a_3 receives its maximum utility of 1.822. Then, a_3 is removed from $\mathbf{A_u}$. At this point, a_4 is the final unhappy player remaining in $\mathbf{A_u}$, which commences its channel assignment game. Because channels have been already assigned to both its links (e_3 and

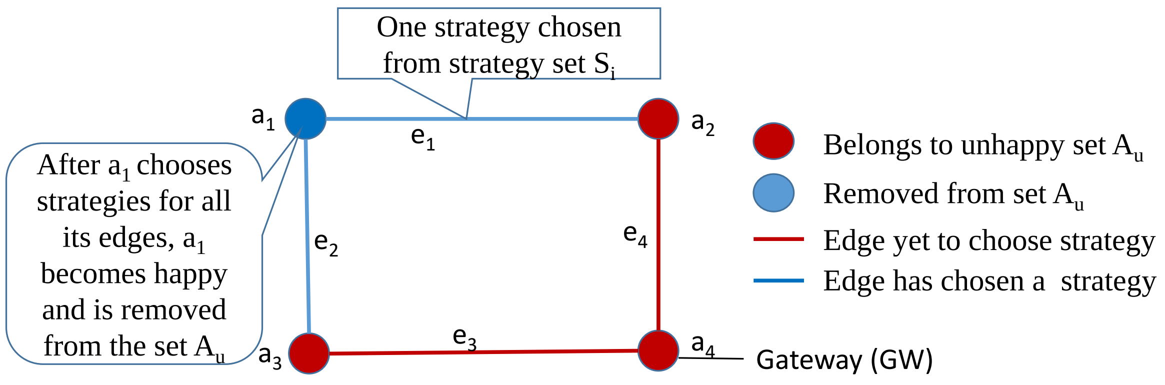

One strategy chosen from strategy set S_i
a1
a2
a3
a4
e1
e2
e3
e4
After a1 chooses strategies for all its edges, a1 becomes happy and is removed from the set A_u
Belongs to unhappy set A_u
Removed from set A_u
Edge yet to choose strategy
Edge has chosen a strategy
Gateway (GW)

Figure 3.1: A simple, easy-to-understand illustration of the operation of the proposed AC-POCA algorithm. The example comprises 4 nodes with 4 links in the proposed network. In this instance, node a_1 is the player with the first initial order. So, it chooses channel assignment strategies on its links (edges) before the other players.

e_4), a_4 does not need to carry out further channel assignment and receives its maximum utility of 1.443. Thus, the total network utility (U_{NET}) with this initial order in row 1 of Table 3.4 is 6.529. In the same manner, the other initial orders lead to different channel assignment on these links resulting in other steady states with varying U_{NET} values.

3.2.3 Dynamic topology

Furthermore, in our combined UAV-enabled IoT, the network topology may dynamically change due to the distance changed between the nodes (UAV or IoT devices), new nodes arrival, and old nodes departure. When the network topology changes, how the game can reach a new steady state should also be considered. In such a case, we focus on the distance df between the nodes. The node arrival and departure are also special cases of distance change. Here, we define df_n as the distance between node a_i and a_{i+1} during the

Table 3.4: Channel assignment using AC-POCA in the different links shown in Fig. 3.1 for different initial orders of the players. Among the 24 possible initial orders, only a few are listed as a simple example.

Order	*Ch (e1)*	*Ch (e2)*	*Ch (e3)*	*Ch (e4)*	U_{Net}
Order:1,2,3,4	1	6	∅	7	6.529
Order:2,1,3,4	1	7	∅	6	6.529
Order:3,4,1,2	6	1	6	11	5.693
Order:4,1,2,3	11	6	1	6	5.693
…	…	…	…	…	…
Order:4,3,2,1	1	7	1	6	5.693

time-slot n. So, the different strategy is chosen by the difference from df_{n-1} to df_n. Those different strategies are considered within lines 4 to 9 of Alg. 1. The lines 4 and 5 show the case of a new node joining the network, lines 6 and 7 show a case whereby the distance between two nodes are far enough to disrupt/break the link. Lines 8 and 9 show the case where the interfere range is changed by the distance change between the two nodes. Then, the steps of mixed channel assignment shown within lines 12 to 23 are repeated so that until none of the nodes belongs to $\mathbf{A_u}$. In this algorithm, each node performs this algorithm by itself. Only when the assignment steps are finished, the broadcast to notify other nodes is performed. Unlike the conventional channel assignment algorithm, nodes calculate their utilities and perform assignment only by local information, except the I-matrix update phase. As shown in the case of the mixed topology, each mixed initial order leads to a unique steady state and the node only performs channel assignment once. This implies that the joining node in the unhappy queue will not interfere with others in front of it. When the environment of the node is changed, the node just pushes itself into the unhappy queue again. This means that even the node reassigns its channel, it will not interfere with the channel assignment of other nodes. Compared with the conventional channel assignment algorithm whereby all nodes should reassign their channels when the topology of the network is changed which causes network transmission to be totally halted, our AC-POCA algorithm can significantly improve the throughput of the network.

Next, we need to demonstrate that for a given initial order of the nodes, the proposed AC-POCA algorithm has a unique steady state. The following section analyzes the uniqueness of the steady state for a given initial order of nodes.

3.3 Analysis on Unique Steady State

From above section, we already know the existence of a steady state (i.e., NE) in our game. To further prove its uniqueness, we use several definitions as follows. As mentioned in Sec. 3.2, we consider our game as a N-player game, $\mathbf{G} = (\mathbf{S}, \mathbf{M})$, where $\mathbf{S}$ is the common pure strategy space $\mathbf{S} = \mathbf{S_i}, \forall i$. $\mathbf{M}$ represents the utility matrix and $\mathbf{x}$ means a selection probability of any strategy in the strategy space, $\mathbf{S}$. $\mathbf{supp}(\mathbf{x})$ indicates the support function of x. When a player chooses strategy x, $\mathbf{br}(\mathbf{x})$ and $\mathbf{wr}(\mathbf{x})$, respectively, represent the sets of the best and worst responses to $\mathbf{x}$ in pure strategy.

From [111], it is known that the Anti-Coordination game has the property that $\mathbf{wr}(\mathbf{x}) \in \mathbf{supp}(\mathbf{x})$ is always satisfied for any strategy $\mathbf{x}$. And from the definition in [112], it may be noticed that the strategy x makes the game reach a NE only when it satisfies $\mathbf{supp}(\mathbf{x}) \in \mathbf{br}(\mathbf{x})$. This means that if any other NE exists, the following equation should

Algorithm 1 Anti-coordination Game-based Partially Overlapping Channel Assignment (AC-POCA) Algorithm.

```
Input: Each node $a_i (i \in 1, N)$ in $\mathbf{A}$
Output: The selected strategy $\mathbf{s_i^{sel}}$ of each link
 1: Initialization: unhappy set $\mathbf{A_u} \leftarrow A$
 2: Set a priority order to every node in set $\mathbf{A_u}$
 3: for each time slot $n$ do:
 4:     $\forall a_i \in \mathbf{A}$
 5:     $qt \leftarrow 0$
 6:     if $df_{n-1} = \infty$ and $df_n < \infty$ then
 7:         Put $a_{i+1}$ into set $\mathbf{A}$ and $\mathbf{A_u}$
 8:     else if $df_n = \infty$ then
 9:         Put $a_i$ and $a_{i+1}$ into set $\mathbf{A_u}$
10:     else if $|df_n - df_{n-1}| > d_h$ then
11:         Put $a_i$ into set $\mathbf{A_u}$
12:     end if
13:     while $|\mathbf{A_u}| \neq \emptyset$ and qt $= 0$ do
14:         Select the first order node $a_{fo}$ in unhappy set $\mathbf{A_u}$
15:         if $a_{fo} = a_i$ then
16:             for each link $\mathbf{e_i}$ of node $a_i$ do
17:                 $\mathbf{s_i^{sel}} \leftarrow$ first strategy in $\{k_{i,1}, \ldots, k_{i,c}, \ldots, k_{i,|C|}\}$
18:                 while $\mathbf{s_i^{sel}}$ does not satisfy (3.7) do
19:                     $\mathbf{s_i^{sel}} \leftarrow$ next strategy
20:                 end while
21:                 if $\mathbf{s_i^{sel}} \neq$ valid strategy($M_i < h_v$) then
22:                     $\mathbf{s_i^{sel}} \leftarrow \emptyset$
23:                 else
24:                     $\mathbf{s_i^{t+1}} \leftarrow \mathbf{s_i^{sel}}$
25:                 end if
26:             end for
27:             $qt \leftarrow 1$
28:             Broadcast $qt$, $ID_{a_i}$ and $\mathbf{s_i^{t+1}}$, and all nodes update I-Matrix
29:         end if
30:         if all link $\mathbf{e_i} = \emptyset$ then
31:             Remove $a_i$ from $\mathbf{A}$
32:         end if
33:         Remove $a_{fo}$ from unhappy set $\mathbf{A_u}$
34:     end while
35: end for
```

be satisfied:

$$\mathbf{wr}(\mathbf{x}) \in \mathbf{supp}(\mathbf{x}) \in \mathbf{br}(\mathbf{x}). \tag{3.11}$$

This implies that the sets of $\mathbf{wr}(\mathbf{x})$, $\mathbf{supp}(\mathbf{x})$ and $\mathbf{br}(\mathbf{x})$ are equal, and each of them is equal to $\mathbf{S}$. However, this is impossible for the strategy in the interior. Thus, the NE, i.e., the steady state is unique in our proposed AC-POCA algorithm.

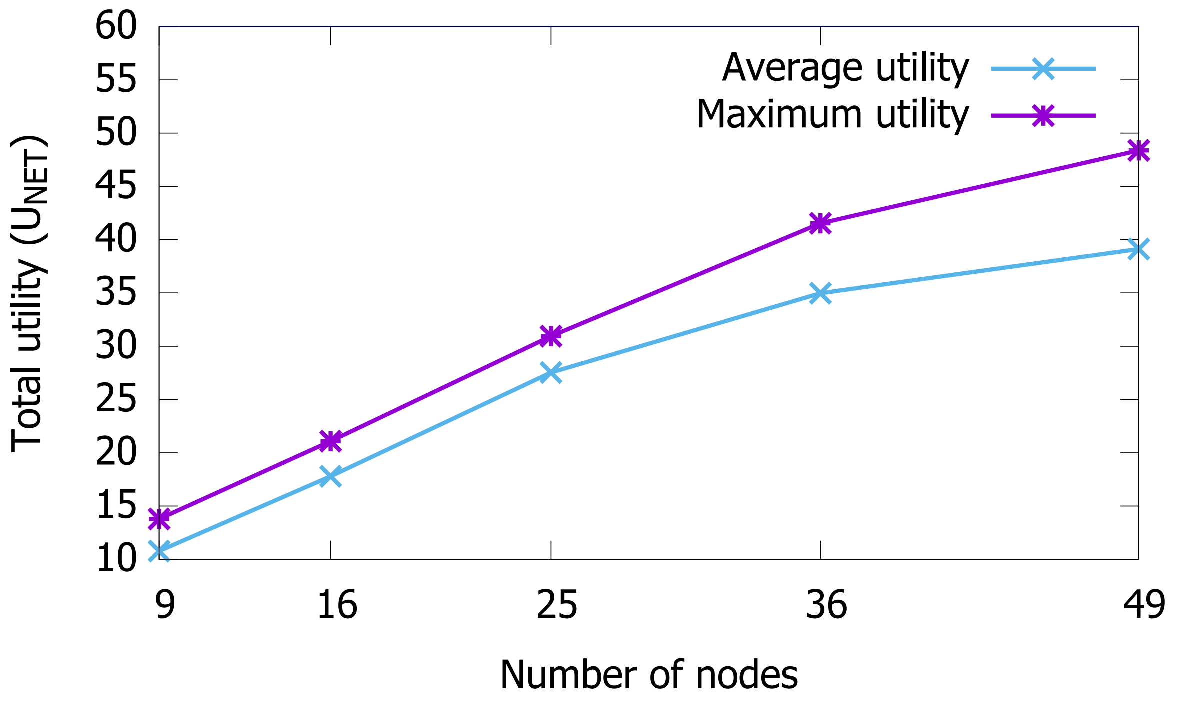


Figure 3.2: The utility function (U_{NET}) of AC-POCA algorithm demonstrating the comparison of maximum and average utilities.

3.4 Performance Evaluation

In this section, we evaluate our proposed AC-POCA algorithm by first analyzing the Price of Anarchy (PoA) to derive the upper bound. Then, computer-based simulation results are provided to further verify the effectiveness of the proposal.

3.4.1 Price of Anarchy (PoA)

The PoA measures how the efficiency of a system is degraded because of the selfish behavior of its agents or players [113]. The PoA measure can be extended to diverse systems including game-theoretic models. In our proposed AC-POCA algorithm, the players (i.e., nodes) can be trapped at a local optimum point where none of the players is willing to change strategy even if the system performance is still distant from the desirable global optimum. Because of this, the efficiency of our game-theoretic algorithm needs to be evaluated through PoA analysis. According to its definition, PoA may be expressed as follows:

$$\text{PoA} = \frac{\max U_{NET}(\mathbf{\Psi}')}{\min U_{NET}(\mathbf{\Psi}'')}, \mathbf{\Psi}', \mathbf{\Psi}'' \in \text{NE}. \tag{3.12}$$

In our earlier work in [29], we demonstrated that the worst and best NEs are the two situations of common channel assignment and non-interfering links channel assignment. Thus, with the definition of PoA, its upper bound is further expressed as follows.

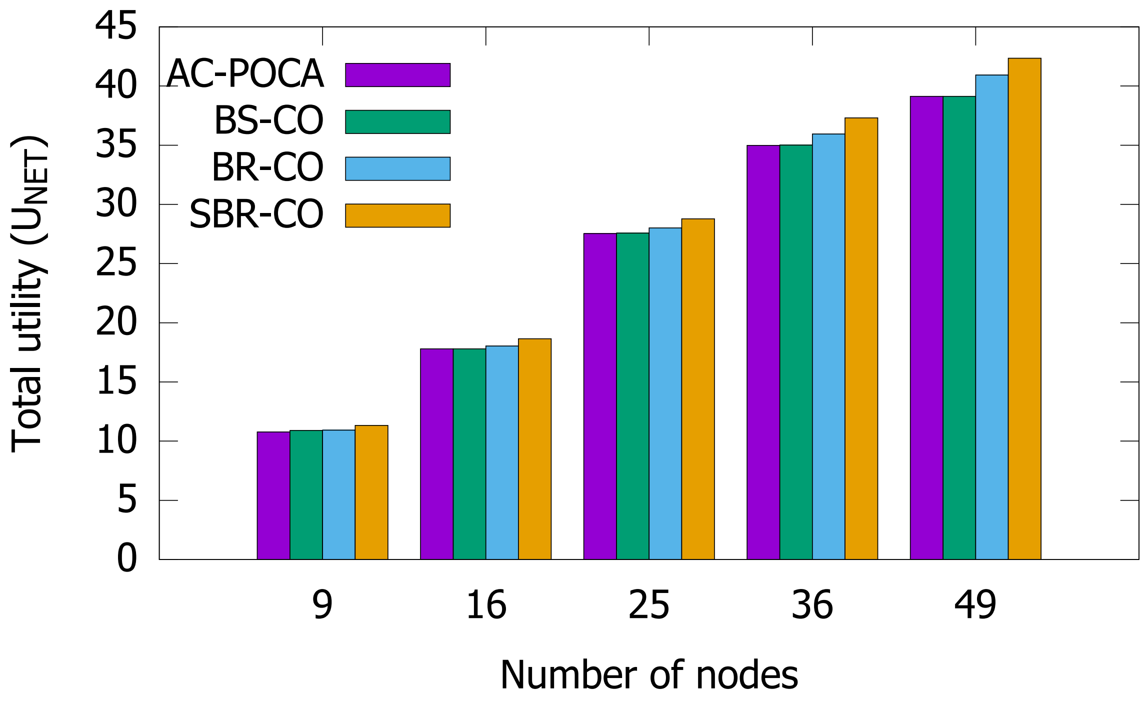


Figure 3.3: The utility function (U_{NET}) of AC-POCA compared with that of a cooperative game with three learning schemes.

- The worst NE for Multi-Radio Multi-Channel (MRMC) networks is the Common Channel (CC) assignment: $U_{NET}^{CC}(\mathbf{\Psi})$.
- The best NE for MRMC networks is a topology with Non-Interfering (NI) links and hop count is the Shortest Path (SP): $U_{NET}^{NI-SP}(\mathbf{\Psi})$.

Thus, PoA of (3.12) can be rewritten as follows.

$$\text{PoA} = \frac{U_{NET}^{NI-SP}(\mathbf{\Psi})}{U_{NET}^{CC}(\mathbf{\Psi})} \tag{3.13}$$

In the remainder of the section, computer-based simulation results are provided to further verify our analysis.

3.4.2 Simulation Results

Our conducted simulations consider two scenarios, the static and dynamic topologies. These simulation scenarios are configured using C++ as follows. First, consider the static topology, a grid topology similar to that described in [29] is constructed as the wireless IoT system. The distance between the neighboring nodes in the network is set to 120 m. The gateway node is positioned at the edge of the simulated grid that is the farthest from the user devices. The nodes are assumed to be equipped with multi-channels, multi-radios operating with IEEE 802.11g wireless technology. For simplicity, the link data rate is set to 8Mbps. In our conducted simulations, the numbers of nodes are varied in the range of {9, 16, 25, 36, 49}.

In Fig. 3.2, we simulate our algorithm in different network settings. Here, we set a time based random seed to initialize the order of the players, and repeat the simulation 1000 times to calculate the average utility. It can be noticed from the plot in this figure that the average utility is quite close to the maximum utility of the network. This corroborates our PoA analysis in (3.13).

Furthermore, we compare the utility of AC-POCA with the cooperative game used in [29] with three learning schemes. For comparison, we use the same player utility function of eq. (3.2). Here, the three learning schemes of cooperative game are referred to as BS-CO (Best Response), BR-CO (Better Response) and SBR-CO (Smooth Better Response). Then, the result is shown in Fig. 3.3. From the result, we can see that the utility of AC-POCA is almost the same as BS-CO, and slightly differ from BR-CO and SBR-CO. To compare with the increased iteration times N_{iter} and significant signaling overhead in eq. (3.10), the slightly lower utility of AC-POCA can be considered as a tradeoff with its significantly lower number of iterations and signaling overhead, which is discussed next.

Now, we compare the performance of our proposed AC-POCA algorithm in the static network topology with two existing methods from [29], i.e., Cooperative Channel Assignment Game (CoCAG) with Best Response (BR) and Smoothed Better Response (SBR). For ease of representation, the two compared methods are referred to as CoCAG-BR and CoCaG-SBR, respectively. The comparison is performed in terms of the convergence time performance and signaling overhead.

Next, we compare the convergence speed of AC-POCA and existing CoCAG-BR and CoCAG-SBR algorithms in Fig. 3.4. In the conducted simulations, we set the factor of finalization criteria in SBR to 85 percent of the maximum utility, which was estimated by a centralized brute force algorithm. The results in the plot in Fig. 3.4 demonstrate that the proposed AC-POCA method has the fastest convergence speed because it uses the unhappy queue and only exploits the local information available to the nodes.

In order to evaluate the signaling overhead of the proposed AC-POCA in contrast with the other two existing methods, we set the length of signaling packets to 1Kb and consider that the nodes use flooding to broadcast the notification qt and their channel selection changes. Fig. 3.5 demonstrates that the signaling overhead is significantly lower with the proposed AC-POCA algorithm compared with that of CoCAG. This is because of the improvement in AC-POCA in terms of convergence speed and optimization of the SBR/BR functions as analyzed earlier section.

After channel assignment is performed to a link, the link is set to an active to verify if

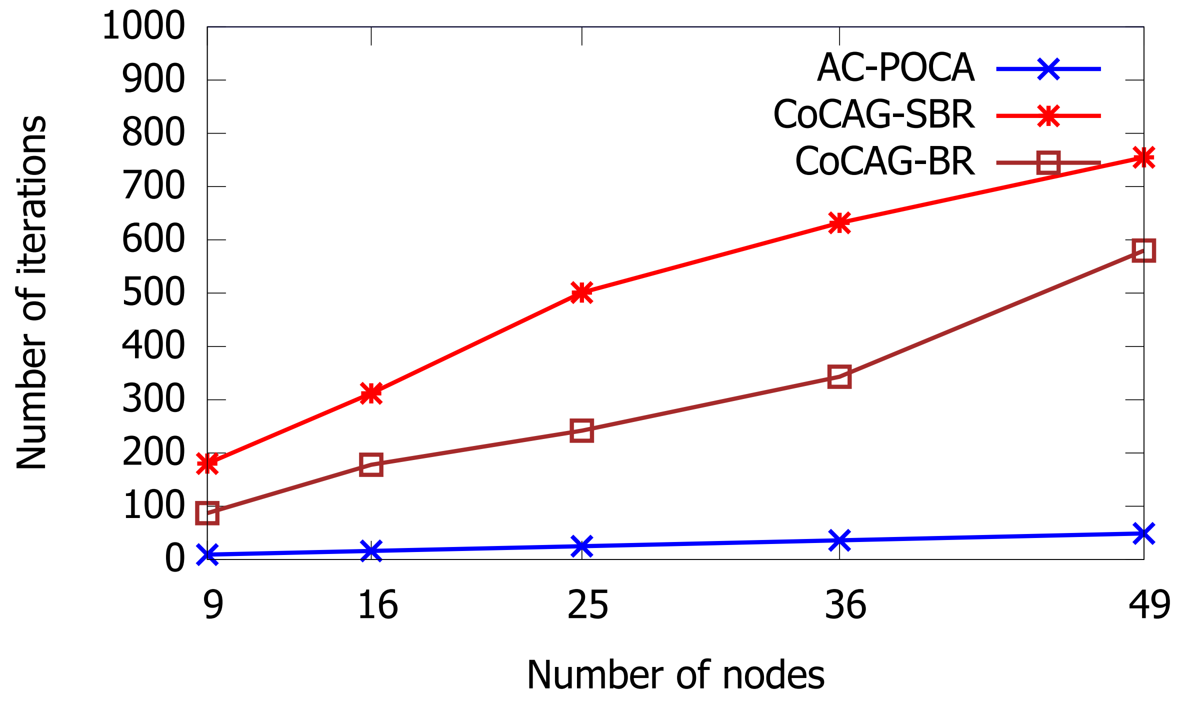


Figure 3.4: Comparison of convergence time performance for the proposal (AC-POCA) and conventional CoCAG algorithms with SBR and BR learning techniques.

the assigned channel is, indeed, viable. In the wireless IoT system, the number of active links directly impacts the aggregate throughput. Therefore, in Fig. 3.6, we compare the number of active links in the proposed AC-POCA algorithm with those achieved by two existing heuristic methods, i.e., the original POC assignment approach [76] and the conventional OC assignment approach. In different network settings, the proposed AC-POCA exhibits the best performance in terms of the number of active links.

Finally, we consider the throughput in the dynamic network topology. Based on the above static scenario, we consider the nodes in the network can randomly move. For simplicity of the conducted simulation, both the data generation and node movement are treated as Poisson processes. We set the moving speed as randomly form 10 to 100 meters/s. The node movement time is set randomly, and the movement rate is set to 1/60s. This means that the node may randomly move from 10 to 100 meters every minute. In such a dynamic scenario, we compare the throughput of AC-POCA and CoCAG which reassign the channels when the topology is similarly changed. The result is demonstrated in Fig. 5.8, in which the process time is 1000s. The data generation rate is 125KB/s in each node. To simplify the simulation, we set the value of h_v, in AC-POCA, in each scenario, as the largest one (only when $IF_j = 0$, no interference exists). From the result, it can be noticed that with the growth of the network size, the throughput performance of our proposal is much better than conventional one. This is because of both channel

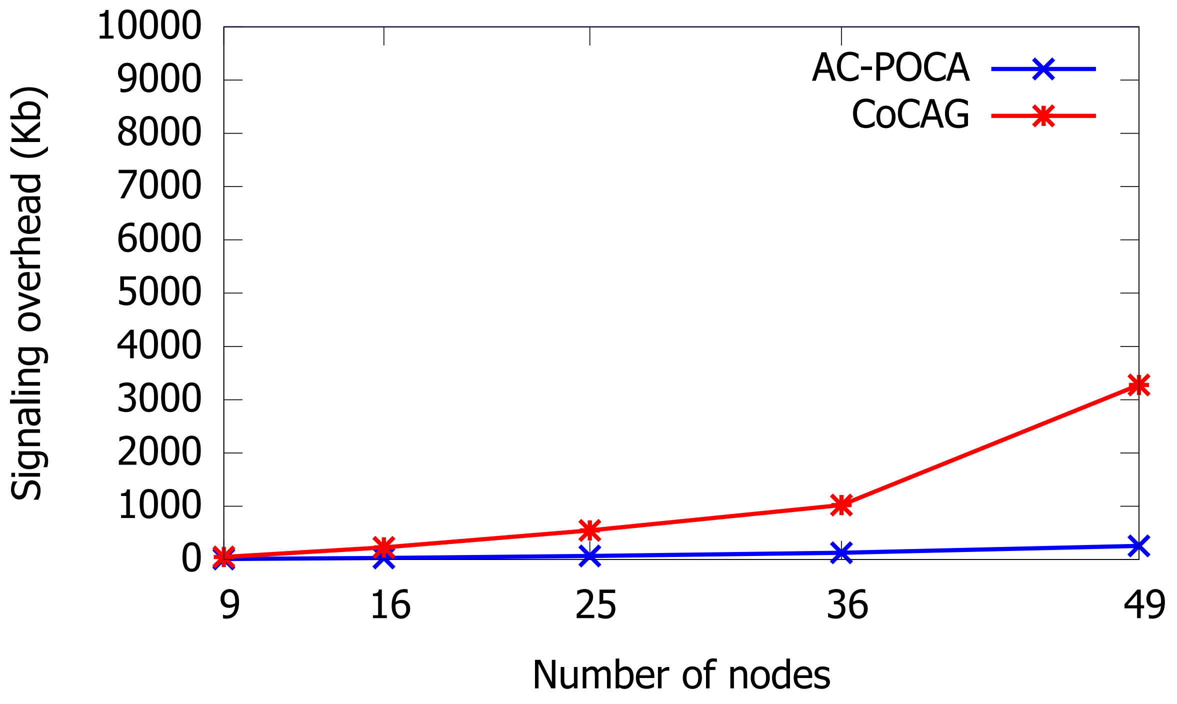


Figure 3.5: Signaling overhead comparison for the proposed AC-POCA and conventional CoCAG (SBR) methods.

assignment time and iteration time of the conventional channel assignment algorithm become much larger when the number of user increases significantly, as shown in Fig. 3.4.

3.5 Summary

The considered UAV-enabled IoT require quick network access. However, quick and dynamic network formulation is required to support such a system. Besides, the radio channel assignment of the nodes is an optimization problem since the number of orthogonal channels is limited and using overlapping channels in adjacent nodes with both primary cellular and D2D links leads to severe interference in IoT. Furthermore, in the considered UAV-enbaled IoT network, the mobility of nodes leads to highly dynamic situation which renders conventional channel assignment algorithms unsuitable. For overcoming those challenges, we presented an interference model and formulated a formal problem. Then, we proposed AC-POCA, a distributed Anti-Coordination game based algorithm for solving the channel assignment problem in the considered UAV-enabled IoT network. Using AC-POCA, the nodes are able to use only local information to reach a steady state in the network. Through analysis, the uniqueness of the steady state in the proposed AC-POCA was also verified. In addition, simulation results were provided to demonstrate that the proposal leads to fast convergence, low signaling overhead, and improved throughput in

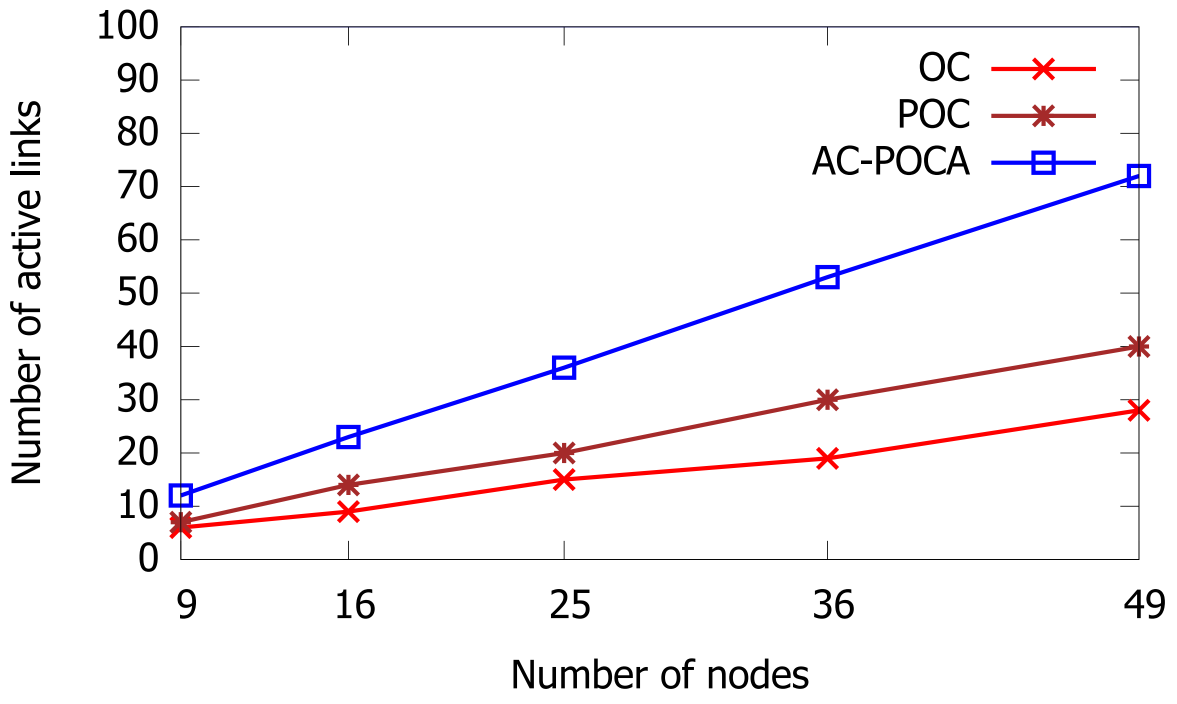


Figure 3.6: Connectivity of network in terms of number of active links in case of the proposed AC-POCA, and conventional OC and POC methods.

contrast with the existing methods.

The significant performance of AC-POCA is based on two assumptions. The first one is the heterogeneous devices in IoT can efficiently connect with each other immediately, the second assumption is that the traffic loads in the IoT are smooth and not suddenly change in ashort time. However, in the real IoT environment, the devices in IoT might be various and not controlled in distributed way whereby the heterogeneous devices and structure lead to more dynamic even bursty network traffic. In next chapter, to deal with the real challenge in IoT, we consider the SDN to make the distributed heterogeneous infrastructure of IoT to a centralized architecture referred as SDN-IoT. Based on the intelligent deep learning, a corresponding deep learning based partially overlapping channel assignment algorithm (DLPOCA) is proposed in the SDN-IoT.

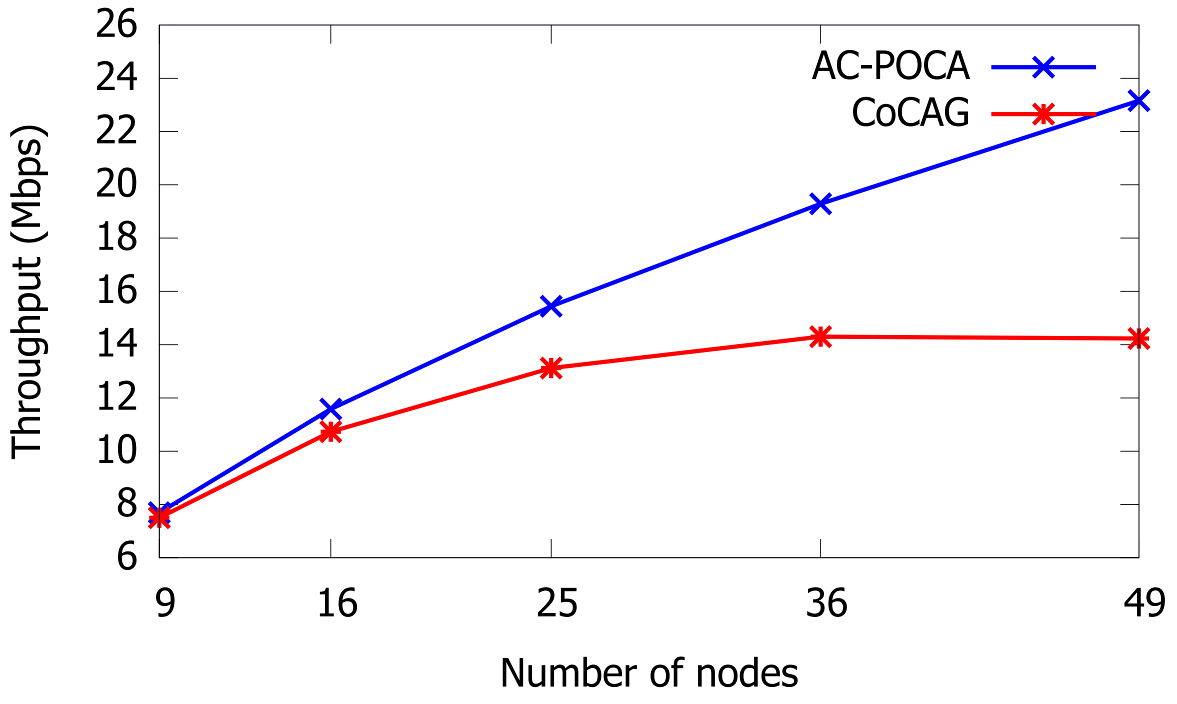


Figure 3.7: Throughput comparison for the proposed AC-POCA and conventional CoCAG (SBR) methods for the dynamic topology.

Chapter 4

Deep Learning Based Traffic load prediction

In previous chapter, we proposed a distributed POC assignment algorithm to solve the radio resource allocation problem in UAV-enabled IoT. Such as the UAV-enabled IoT scenario, due to the high mobility and wide coverage of these devices, different types of wireless radio access technologies like cellular, MEC and D2D have been widely used in IoT. The complexity of heterogeneous communication technologies and device infrastructures have resulted in many critical issues, such as the task and space sharing among different devices, the network load balance, and so forth. Therefore, the assumptions used in the previous scenario may not practical in the real heterogeneous IoT.

As described in introduction, to better suit the heterogeneous large scale IoT, the Software Defined Networking (SDN) [33] technology has been proposed as a novel solution to connect the distributed heterogeneous devices into a centralized sharing working system. This is referred as the SDN-IoT [35, 50]. In SDN-IoT, as shown in Fig. 1.2, various devices includes UAVs are widely deployed in the sensing plane. All sensing data collected by the sensing plane are forwarded through switches in data plane and then delivered to the gateway. Using the control plane, SDN-IoT separates the network control logic from the underlying routers and switches to the central controller, which usually has high computation capacity. Thus, the controller is able to control the whole network, e.g., by computing packets forwarding paths and managing the channel resource, while the switches in the data plane are just responsible for forwarding the massive IoT data.

The wireless SDN-IoT meets the requirement that huge number of heterogeneous devices work cooperatively in one large scale network. And the state information from devices can be collected through the SDN network immediately. However, with the increasing number of devices, the traffic load of switches may become significantly heavy, and multiple channels need to be appropriately allocated to links. In addition, heterogeneous devices have different policies in data sensing and collection, which result in uneven

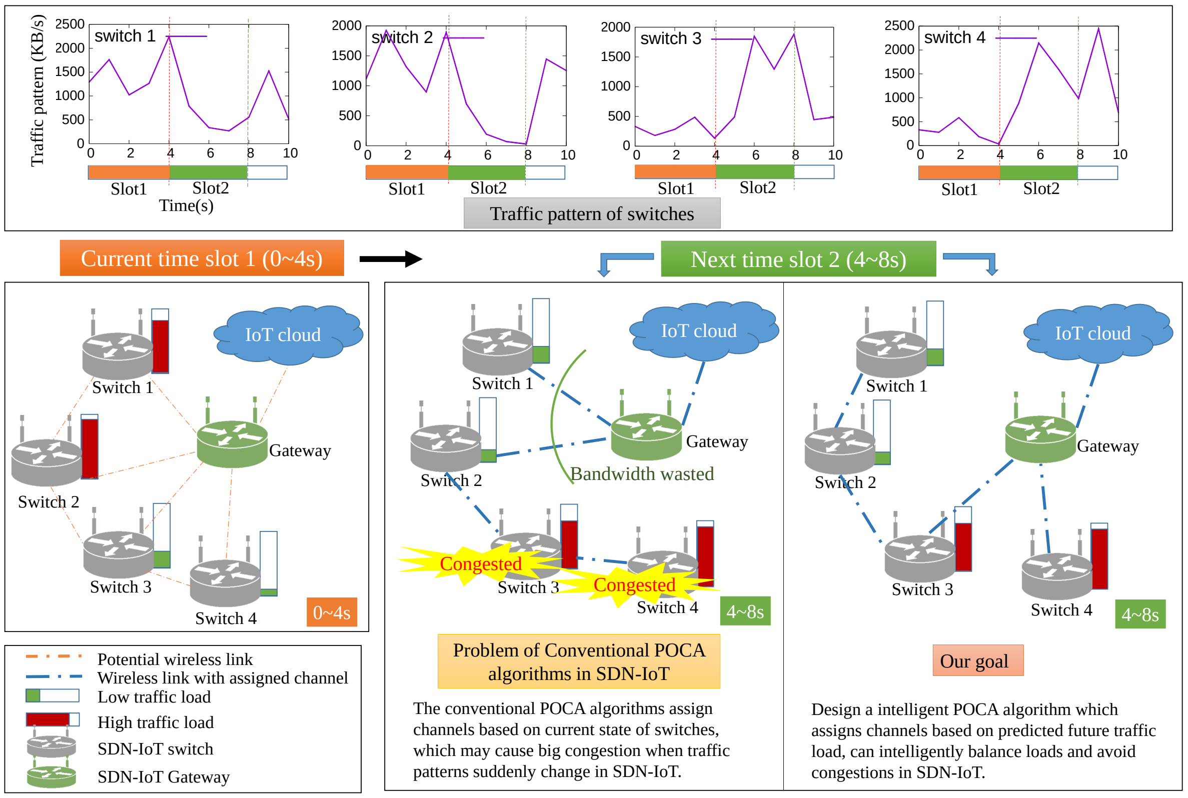

Traffic pattern (KB/s)
switch 1
switch 2
switch 3
switch 4
Slot1
Slot2
Time(s)
Traffic pattern of switches
Current time slot 1 (0~4s)
Next time slot 2 (4~8s)
IoT cloud
Switch 1
Switch 2
Switch 3
Switch 4
Gateway
0~4s
4~8s
Bandwidth wasted
Congested
Problem of Conventional POCA algorithms in SDN-IoT
The conventional POCA algorithms assign channels based on current state of switches, which may cause big congestion when traffic patterns suddenly change in SDN-IoT.
Our goal
Design a intelligent POCA algorithm which assigns channels based on predicted future traffic load, can intelligently balance loads and avoid congestions in SDN-IoT.
Potential wireless link
Wireless link with assigned channel
Low traffic load
High traffic load
SDN-IoT switch
SDN-IoT Gateway

Figure 4.1: The problem of existing POC assignment algorithms and our research goal.

bursty traffic arrival of switches. For such situations, how to adaptively assign channels to fit such bursty traffic becomes a significant research challenge.

As depicted in Fig. 4.1, conventional POC assignment algorithms [29, 105, 75, 29, 114] and even our ACPOCA algorithm only focus on the current (i.e., last time slot) traffic load, which works well with the assumption of stable traffic loads. However, once the traffic pattern suddenly changes in the next time slot, the channel in good condition may be assigned to a wrong link with a heavy load in the last time slot, but idle in next one. On the other hand, the link with high load in the next time slot may be assigned a channel in poor condition due to its idle state in the last time slot. The wrong channel assignment decision significantly wastes the channel resource, and this leads to decreased network throughput and high packet loss rate.

Furthermore, conventional POC assignment algorithms do not consider the dynamics of traffic patterns and perform the channel management in a static manner. Thus, they carry out the channel assignment only once in the initial part in a distributed fashion, and have the problems of high computation complexity and long iteration time. In a dynamic IoT environment, the channels need to be reassigned once the network traffic condition changes. However, when the channel assignment is being processed, the network transmission must be suspended until new channels are available. Thus, the high computation complexity and long iteration time of conventional algorithms may lead to long suspension time of network transmission.

Therefore, in order to improve the network transmission performance, two main problems need to be solved. One is the dynamic traffic load prediction problem, and the other is the problem of how to achieve the quick convergence of channel assignment algorithm to reduce transmission suspension time.

In this vein, a deep learning based intelligent POC assignment algorithm is proposed. Our proposal consists of two parts. First, we utilize deep learning technique to predict the future traffic loads of switches according to the history of traffic data. Then, the central controller of SDN-IoT can further adopt the deep learning technique to allocate the channel resource according to the traffic load prediction. The centralized control mechanism in SDN-IoT can ensure the traffic load prediction accuracy, while the high computation ability of the central controller in SDN expedites the POC assignment process.

To better describe our proposal, we at first model the network of SDN-IoT in Sec. 4.0.1, the used deep learning model is detailed described in Sec. 4.1. Then, based on the control manner of the IoT, we proposed three network traffic prediction mechanisms based on deep learning in this chapter. After the prediction, the deep learning based partially overlapping channel assignment algorithm (DLPOCA) and the enhanced traffic prediction based DLPOCA referred as TP-DLPOCA are proposed in next chapter.

4.0.1 Network Model

Consider the SDN-IoT is constructed in a heterogeneous structure which contains different kinds of devices. Devices sense and collect data, and then send the data to the gateway through multiple switches. For better understanding, we use graph $\mathbf{G} = (D \cup S \cup C, E)$ to represent the network where D denotes the set of devices in the network and $D = \{d_1, d_2, \ldots, d_{|D|}\}$. And S denotes the set of switches and $S = \{s_1, s_2, \ldots, s_M\}$ where M is the total number of switches. Consider the switches are randomly deployed in the considered area, and each switch serves the devices located in its own service area. For example, an Access Point (AP) of a residence is regarded as a switch and all the devices in this house are served by the AP. The average number of devices belong to each switch area is presented as R, namely $|D| = M \times R$. Each switch collects data from devices, and then send them to the gateway with multi-hop transmission. The central controller C is deployed randomly in the network as the global network viewer to manage all packets forwarding, deep learning process, channel assignment and other network problems. The structure of the SDN-IoT is shown as Fig.1.2.

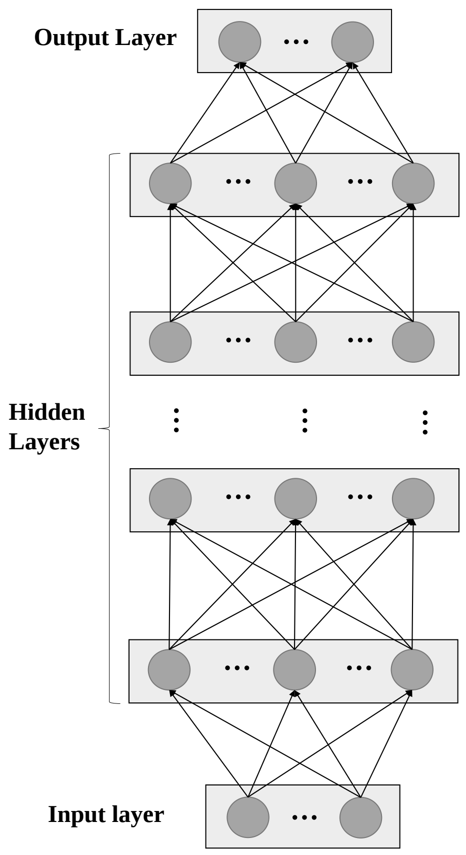
In the sensing plane, we consider Q different kinds of periodic sensing devices and W different kinds of event driven sensing devices with totally number of $|D|$ deployed in the whole area. For example, one kind of periodic sensing device senses and collects 10kB data in every 30s and another kind of periodic sensing device collects 7kB data in every 20s.

Let E represents the edges set in the graph $\mathbf{G}$. Furthermore, the edge $e \in E$ in the graph means the link between two vertices. The weight, $w(e)$, represents the connection ability of the link e. This weight depends on many factors such as the transmission distance, transmission power, interference, bandwidth, and so on. Consider the links between devices and switches use different spectrum from the links between switches. The data sensed by a single device are small and the capacity requirement of a single link between devices and switches is not so strict. Therefore, the considered interference mainly exists in the links between the switches in the data plane.

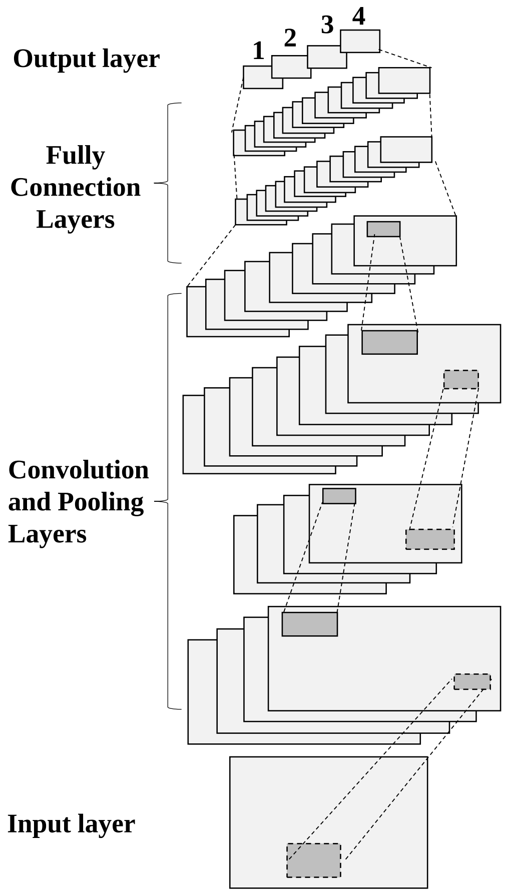
The interference already introduced in Chapter. 3. In order to quickly measure all the channels conditions, in the conventional partially channel assignment algorithms, each router uses the interference matrix ($IMatrix$) to record the $f_{p,q}$ value of all the links. And all routers need to broadcast their channel information and update $IMatrix$ continuously, which result in large signaling. On the other hand, in our proposed deep learning based channel assignment algorithm, the $IMatrix$ is no longer needed, each switch just receives traffic load information and activates neural network weight matrix obtained by the training process. The only signaling overhead is the traffic load transmission process between switch and central controller. Next, we describe the deep learning training model used in training process.

4.1 Deep learning model

In our proposed training process, we consider two neural network structures, the basic Deep Belief Architecture (DBA) and the deep Convolutional Neural Network (CNN). As shown in Fig. 4.2a, the chosen DBA is constructed with L layers, including one input layer, one visible output layer and $(L-2)$ hidden layers. The unit in each layer except the input layer has its own weight value called bias. And the units in two adjacent layers are connected with each other via weighted links while no inner layer connection exists. Let x_{input} and y_{output} denote the values of units in the input and output layer, respectively. w_{ij} denotes the weight of link between units i and j, and b_i represents the bias of unit i. Additionally, w and b represent the matrices consisting weights of all links and all the bias values, respectively. The training of the DBA consists of two steps, namely forward propagation and back propagation processes. The forward propagation is used to construct the structure and activate output, while the back propagation is used to adapt the structure and fine-tune the values of w and b. As modeled in our previous work



(a) Considered DBA structure.



(b) Considered deep CNN structure.

Figure 4.2: The employed deep learning structures.

in [103], the forward propagation process can be modeled as a log-likelihood function,

$$l(w, b, x_{input}, y_{output}) = \sum_{t=1}^{m} \log p(\boldsymbol{v}^{(t)}), \tag{4.1}$$

where, $\boldsymbol{v}^{(t)}$ denotes the t^{th} training data. The DBA training process can be seen as a log-linear Markov Random Field (MRF). Hence, we use $p(\boldsymbol{v}^{(t)})$ to represent the probability of $\boldsymbol{v}^{(t)}$. Here, m represents the total number of training data.

Since the purpose of the training process is to maximize $l(w, b, x_{input}, y_{output})$, in the backpropagation process, the gradient descent method is adopted to adjust the link weight w and bias b, which is represented as:

$$w = w + \eta \frac{\partial l(w, b, x_{input}, y_{output})}{\partial w}, \tag{4.2}$$

$$b = b + \eta \frac{\partial l(w, b, x_{input}, y_{output})}{\partial b}, \tag{4.3}$$

where, η is the learning rate of training process.

The second considered deep learning structure is the deep CNN as shown in Fig. 4.2b. At the first glance, the structure of deep CNN is similar to DBA, and the main training process also includes forward and back propagation. However, when the size of the input layer becomes quite large and spatially connected in high dimensions, the DBA cannot capture the spatial features efficiently. As a powerful deep learning structure, the deep CNN is widely used in image identification and natural language processing [115, 116]. In the deep CNN, the covolutional layers are good at capture the spatial and temporal connections of the input data.[117]. This is a better choice to construct the learning system of centralized network of spatial connection extraction. To better extract the spatial connections of input data, the convolutional and pooling layers are employed in the deep CNN. The convolution operation is used to filter the input and pass the result to the next layer, while the pooling layers are used to combine the outputs of the neuron clusters at one layer into a single neuron in the next layer, which can further reduce the redundant data and extract the wide range spatial features. Different filters may be used in each convolutional layer and their results are combined to transfer to fully connection layers. With the utilization of convolutional and pooling layers, the features of input can be efficiently extracted, which significantly reduces the computation burden.

As the purpose of the convolution operation is to extract the distinguished features of the input, the parameters (weights and biases) of the convolution operation consist of a set of learnable filters. If we use $W^{(l_1)}$ to denote the filters and the k^{th} filter is represented by $W_k^{(l_1)}$, the obtained feature map by the convolution operation can be shown as follows.

$$\begin{aligned} u_{i,j,k}^{(l_1)} &= (U^{(l_1-1)} * W_k^{(l_1)})(i,j) + w_{bk}^{(l_1)} \\ &= \sum_{p=1}^{P}\sum_{m=1}^{M'}\sum_{n=1}^{N'} w_{m,n,p} a_{i+m,j+n,p}^{(l-1)} + w_{bk}^{(l_1)} \end{aligned} \tag{4.4}$$

$$a_{i,j,k}^{(l_1)} = f(u_{i,j,k}^{(l_1)}) \tag{4.5}$$

where $f(\cdot)$ is the activation function and $a_{i,j,k}^{(l_1)}$ is the activated value of the unit in the i^{th} row and j^{th} column of the feature map. Therefore, $u_{i,j,k}^{(l_1)}$ is the value before activation. $w_{bk}^{(l_1)}$ denotes the bias of the k^{th} filter and is usually a single numeric value. $a_{i+m,j+n,d}^{(l_1-1)}$ is the activated value of unit in the $(i+m)^{th}$ row and $(j+n)^{th}$ column. Besides the convolution layers, the full connection layers are used to construct the basic training structure which is similar to DBA. Then, the similar forward propagation and back propagation processes are repeated to fine-tune the whole CNN structure.

In SDN-IoT, the central controller is the brain to control all functions of switches. All control and computation tasks are handled in the control controller, which is a totally centralized control system. In order to research the performance between using centralized SDN system and semi-centralized or distributed conventional control system without centralized SDN, we separately design our deep learning based traffic load prediction algorithm into three different systems.

In the conventional network, the switch (i.e., router, in order to easily describe, we still simply call a router in conventional network a switch) only knows local information and communication with each other in a distributed manner, which is referred to as a distributed control system. There is also a kind of mixed control system, in which the central controller is deployed with limited computation and communication ability. The limited central controller only knows part of the global information. In such a mixed system, the switches need to handle a part of the tasks in a localized manner and suffer from limited service from the central controller. Such a system can be treated as a semi-central control system. Based on these three different control systems, we propose three deep learning based traffic load prediction methods, namely, Central control based Traffic load Prediction (CTP), Semi-Central control Traffic load Prediction(S-CTP), and Distributed control Traffic load Prediction (DTP). Next, we describe the three prediction methods, respectively.

It is worth mentioning that apart from the link condition of each switch, the main factors influencing the traffic load is the arrival traffic flow. As mentioned earlier, in each switch, the traffic load sequence, TL, consists of two parts: the relayed traffic flow from other switches denoted by TL_rel, and the integrated traffic flow composed by the sensing data from devices in the sensing plane denoted by TL_int. Therefore, $TL = TL\_rel + TL\_int$.

4.2 Traffic Load Prediction in Central Control System:CTP

In the prediction process of central control system, there are four phases, i.e., data collection phase, training phase, prediction phase, and online training phase.

4.2.1 Data Collection Phase

In the central control system, all the information of switches are periodically collected by the central controller. The central controller records the traffic load sequence, TL, of every switch in the last N time slots. And the length of each time slot is represented as Δ. The traffic load of switch i in last time slot k is recorded as tl^i_k. Then, the past traffic loads TL^i of switch i are formed as a length-N vector, $TL^i = \{tl^i_k, tl^i_{k-1}, \cdots, tl^i_{k-N+1}\}$, where N

represents the number of considered past time slots. N depends on the complexity of input data and is decided according to the training performance. In this case, the controller collects all traffic load series of every switch, and formats them as a traffic load matrix $TL = \{TL^1, TL^2, \cdots, TL^M\}$. From the point of the time series, the traffic loads of all switches in the last N time slots can be also represented as $TL = \{tl_k, tl_{k-1}, \cdots, tl_{k-N+1}\}$.

After data collection, the traffic load matrix TL is used as the input of training data. In the next time slot, the central controller records the traffic load as the real future traffic load $tl_{k+1} = \{tl^1_{k+1}, tl^2_{k+1}, \cdots, tl^M_{k+1}\}$ which will be utilized as the output of training data. After thousands of time slots, the central controller collects thousands of such labeled data and adopt those labeled real data to train the deep neural network in the training phase.

4.2.2 Training Phase

In this phase, in order to obtain a better training performance, we use a Deep Convolutional Neural Network (deep-CNN) to fit our matrix based training data[118]. In our earlier work[100], we compared the training performance with different output formats, and the result shows that the complex output significantly impair the training accuracy. In other words, utilizing only one deep-CNN to predict the future traffic load of all switches, which needs to use the full tl_{k+1} as the output, is too resource-consuming and has a significantly low accuracy. Therefore, we decouple the complex of output and use M deep-CNNs, where each deep-CNN is only used to predict the traffic load of one switch. Thus, the central controller only uses the future traffic load of one switch as the output of corresponding deep-CNN. For example, the training data of deep-CNN CNN^i is $(x_{input}, y_{output}) = (TL, tl^i_{k+1})$. Then, the central controller trains all the deep-CNNs, respectively, to obtain all the stable weight matrices.

4.2.3 Prediction and Accuracy Calculation Phase

In the prediction phase, the central controller undertakes the future traffic load prediction and calculates the prediction accuracy. In this phase, the weight matrix of each deep-CNN obtained in the training phase is adopted to predict the future traffic load, which is a forward propagation process as mentioned in Sec. 4.1. The output of all deep-CNNs is recorded as $TLP_{k+1} = \{tlp^1_{k+1}, tlp^2_{k+1}, \cdots, tlp^M_{k+1}\}$. As mentioned above, the real future traffic load of time slot $(k+1)$ is recorded as $TL_{k+1} = \{tl^1_{k+1}, tl^2_{k+1}, \cdots, tl^M_{k+1}\}$. Therefore, we can calculate the prediction accuracy according to the following equation.

$$\frac{1}{K \times M} \sum_{k=\mathbf{0}}^{K-1} \sum_{i=1}^{M} \frac{|tlp^i_{k+1} - tl^i_{k+1}|}{tl^i_{max}}, \tag{4.6}$$

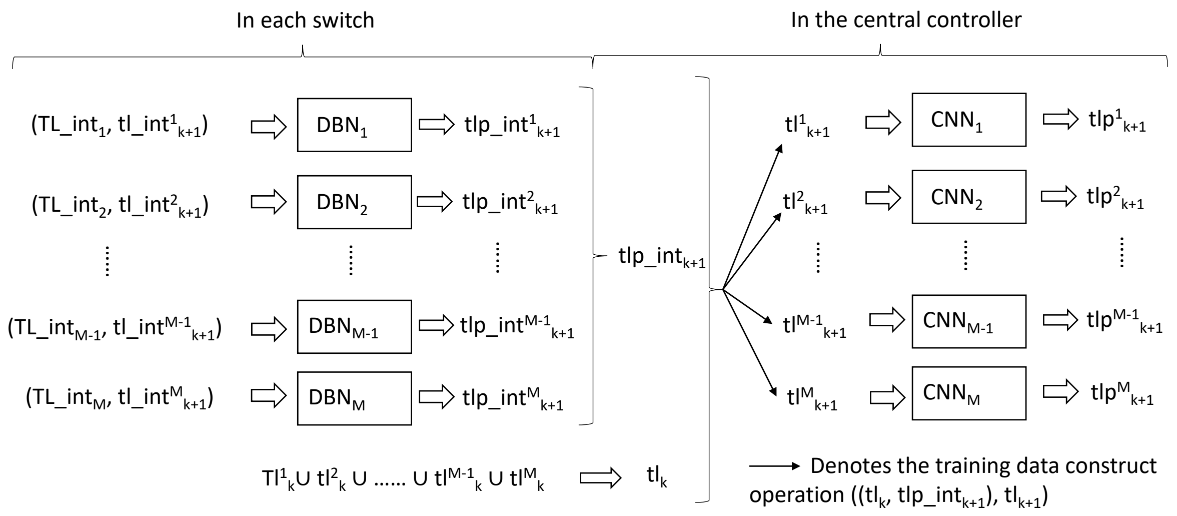


Figure 4.3: The training and prediction phase in a semi-central control system.

where, K represents the total number of considered time slots. tl^i_{max} represents the maximum traffic loads of switch i, Here, we simply consider the maximum traffic load is equal to the maximum buffer size of the switch.

4.2.4 Online Training Phase

If the generation policy of input traffic always acts as a certain pattern, the training and prediction processes, based on only the existing training data, are reasonable. However, in a practical network, the generation policy of the input traffic may change because of some reasons, such as some devices break down, or some new sensing tasks are assigned to existing devices. Based on such situations, the training process should also be adapted correspondingly. Then the online training phase is necessary for adjusting the deep-CNNs to adapt to the new environment.

In this online training phase, each switch continuously records the traffic load data, and the training phase is processed periodically with the collected new training data. Therefore, the weight matrices are periodically adjusted.

4.3 Traffic Load Prediction in Semi-central Control System:S-CTP

In this kind of system, we consider the central controller only has some limited computation ability and the switches need to finish some tasks in a localized manner. In this case, each switch makes some simple pre-prediction just with the local information to alleviate the computational burden of central controller. And the final prediction is still conducted by the central controller with integrated global pre-prediction information from all switches.

4.3.1 Data Collection Phase

In the central control system, the central controller predicts the traffic load based on collected traffic patterns of all switches, that needs highly central computation ability and correspondingly fast communication mechanism of SDN technique. However, with the limited ability of the central controller in a semi-central control system, each switch cannot simply transfer all raw traffic information to the central controller because this will put much burden on the central controller. Thus, in the semi-central control system, switches should perform some pre-treatment of the raw data and send less information to the central controller to decrease both computation and signaling overheads of the central controller. In this case, for each switch i, it records the traffic load tl_k^i of the last time slot, and also separately records the relayed traffic load TL_rel^i and integrated traffic load TL_int^i of the last N time slots. Then, each switch i predicts the future integrated traffic load $tlp\_int_{k+1}^i$ of the next time slot by using recorded TL_int^i as input. This training and prediction process is conducted in the training phase. Then, the switch sends the obtained $tlp\_int_{k+1}^i$ and recorded traffic load of last time slot tl_k^i to the central controller. The central controller collects the data from all switches, and constructs them as the training data tl_k and tl_int_k.

4.3.2 Training Phase

The training phase consists of two steps. The first step is that each switch trains a local neural networks to predict its future integrated traffic load with its past N-time-slot integrated traffic loads. Therefore, for switch i, the training data of its local neural network can be represented as $(x_{input}, y_{output}) = (TL\_int^i, tl\_int_{k+1}^i)$. Since the input is much simpler compared with the input of deep-CNN utilized in the central control system and the training can be treated as the function fitting process between the input and output, here, we can just use deep belief network (DBN or DBA) mentioned in Sec. 4.1 to perform this training process. As we mentioned in the data collection phase, the trained DBA will be utilized to predict the future integrated traffic load which is represented as $tlp\_int_{k+1}^i$, and the results will be periodically sent to the central controller.

When switches finish self prediction and send the result to the central controller, the central controller performs the final prediction with last time slot traffic load tl_k and predicted integrated traffic load tlp_int_{k+1} of all switches. Since the traffic load and integrated traffic are two different network features, we form them as two channels of the input data, similar to our earlier research in [118]. Therefore, the input of training data can be formed as a matrix $(tl_k, tlp\_int_{k+1}) = (\{tl_k^1, tl_k^2, \cdots, tl_k^M\}, \{tlp\_int_{k+1}^1, tlp\_int_{k+1}^2, \cdots, tlp\_int_{k+1}^M\})$. As mentioned earlier, the deep learning structures in the central controller are utilized to predict the future traffic loads of all switches. Similar to the central control based prediction, we utilize M deep-CNNs to make the prediction to alleviate the computational burden and guarantee the accuracy. Therefore, for CNN^i, its labeled training data is formed as $(x_{input}, y_{output}) = ((tl_k, tlp\_int_{k+1}), tl_{k+1}^i)$.

Except the above-mentioned two phases, the prediction phase and online phase in the semi-central control system are almost the same as those in the central control system. The whole training process of each switch and the central controller is shown in Fig.4.3.

4.4 Traffic Load Prediction in Distributed Control System:DTP

In the conventional distributed network, the switches (i.e., router) do not know the global information, and the prediction must be executed in each switch only according to its local information. Thus, a local information based distributed traffic load prediction method is designed as follows.

In the distributed control system, each switch only collects its own traffic load including the relayed traffic load and integrated traffic load. Without additional information of other switches, the relationship between two kinds of traffic loads in different time slots becomes more complex. Therefore, the deep-CNNs utilized in this system are much wider and deeper than the deep-CNNs used in the central and semi-central control systems.

In order to get better training performance, we try two forms of the training data. The first one is to separate the integrated traffic load and relayed traffic load as input. Therefore, switch i records the integrated traffic load TL_int^i and relayed traffic load TL_rel^i of the last N time slots. Then, the two traffic loads are constructed to a two channel matrix as input of training data. Correspondingly, the traffic load in the next time slot, tl_{k+1}^i, is taken as the output. Thus, the training data can be represented as $(x_{input}, y_{output}) = ((TL\_rel^i, TL\_int^i), tl_{k+1}^i)$.

The second kind of input is only the combined traffic load TL^i. In this case, the training data can be denoted as $(x_{input}, y_{output}) = (TL^i, tl_{k+1}^i)$. And in the simulation (presented later in sec. 5.2), we compare the two kinds of training data and find that in current simulated network environment, both methods can achieve the same accuracy (i.e., above 85% in the network with 16 switches.). However, it takes more time for the first method to converge. Thus, we temporarily use the second method as the DTP training data in our research.

The CTP, SCTP and DTP methods are designed to fit the aforementioned three different kinds of control systems. The comparison of the prediction performance with those different methods are researched in section. 5.2.

4.5 Summary

In this chapter, based on the SDN-IoT structure, we proposed three deep learning based traffic prediction algorithms based on different control manner. The used deep DBN and CNN are detailed described in the system model part. From the simulation result, the centralized SDN based control method shows better performance than conventional distributed and semi-distributed manner. After the network traffic prediction, the deep learning based channel assignment algorithm is proposed in the next chapter.

Chapter 5

Proposed Deep Learning Based Partially Channel Assignment

After the traffic loads of the next time slot are predicted by our proposed prediction methods, many existing channel assignment algorithms which are based on the traffic profile can be used to assign proper channel to each link. However, due to the problem we mentioned in Chapter. 1, the conventional channel assignment algorithms with slow convergence cannot meet the new requirement of the considered SDN-IoT. Aided by the high computation ability in the future SDN [103], we propose a new deep learning based channel assignment algorithm, which shows better convergence performance than the conventional algorithms, and leads to better network throughput.

In this chapter, we at first propose a supervised deep learning based partially channel assignment algorithm (DLPOCA). In the DLPOCA, the deep DBN is employed to be the training structure. With the collect data from the previous ACPOCA contains traffic patterns and assigned channel number of each links, the training data are characterized from collected data to formatted tensor. With offline learning, the deep DBNs are fully trained with the training data. Then, through the trained DBNs, the SDN central controller can intelligent made channel assignment decision by inputting the traffic patterns of IoT.

After the DLPOCA, we further combine the DLPOCA and the traffic prediction, propose the traffic prediction based DLPOCA referred to as TP-DLPOCA. In the TP-DLPOCA, instead of the traffic load of current time slot, we use the predicted traffic load and the assigned channel number of next time slot as training data to train and active the trained deep DBNs. The main difference between the TP-DLPOCA and conventional methods is demonstrated in Fig.5.1.

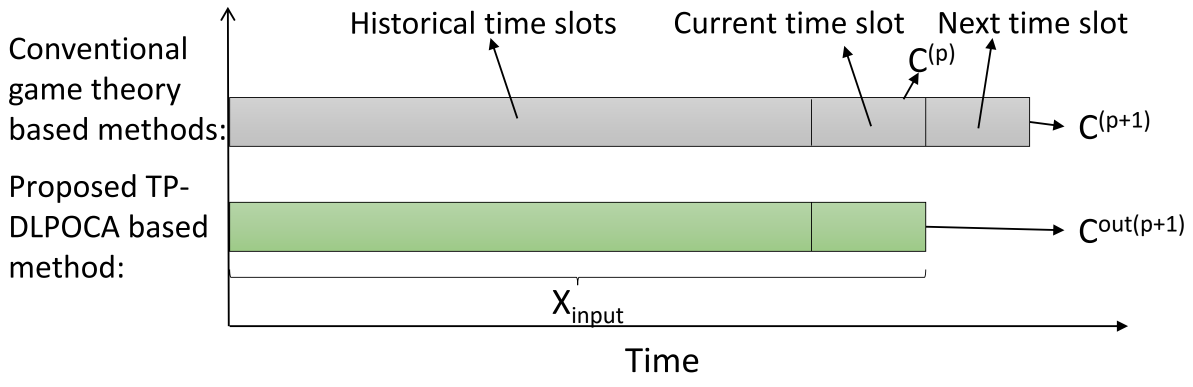


Figure 5.1: The decision process comparison of the TP-DLPOCA and the conventional game theory based POC assignment methods.

5.1 Proposed Deep Learning Based Partially Channel Assignment

In this section, we introduce the proposed Proposed Deep Learning Based Partially Channel Assignment (DLPOCA). In our proposal, we use deep learning to train the network with the data from our prior proposed ACPOCA, in which, the partially overlapping channels are assigned to each link by using an anti-coordination game. In the anti-coordination game based partially channel assignment algorithm (AC-POCA), each router (i.e., switch) chooses the channel of its links by using a utility function and plays game with other switches. Different from cooperative game, AC-POCA can always get a unique stable state in the network, and such uniqueness of AC-POCA makes the algorithm appropriate to be trained by deep learning and gets almost the same accuracy as that of AC-POCA. If we set the AC-POCA as a benchmark, the deep learning based channel assignment algorithm can get 100% accuracy compared with the benchmark. Besides the same channel assignment accuracy, the deep learning based assignment algorithm can save the game process time between switches, leading to much faster convergence.

To train the network, we try to find the main features of training data. In ACPOCA with fixed topology, traffic load is the main feature to order the routers in the queue of game, and the order significantly affects the channel assignment result of each router. In the intuition of human, the traffic load (TL) should be the main feature of training data, and some other features should also be considered such as the hop count (HC) to the gateway and interference factor (IF) of each link. Then, we respectively use those features and some combinations of them to construct the different format of input of training data. In the experiment result shown in Table 6.1, we can find that any feature combination containing the feature of traffic load can get 100% accuracy. To the opposite extreme, the accuracy of using any combination without traffic load is less than 70%.

Besides the accuracy rate of different combinations, we compare the training epochs (i.e., training time) of the combinations containing traffic load. And the result shows that the method only using traffic load as input of training data can get the best performance of training. Therefore, in the training process, the traffic load is utilized as the main feature to construct the input of training data.

Here, we divide the channel assignment algorithm into two parts. In the first part, we propose a Deep Learning based Partially Overlapping Channel Assignment algorithm (DLPOCA). In the second part, we further propose an intelligent deep learning channel assignment strategy which joints the DLPOCA with traffic load prediction algorithm, referred to as TP-DLPOCA, to obtain further improved performance.

5.1.1 Deep Learning based Channel Assignment

To better describe our proposal, we divide the whole assignment process into two steps, i.e., the training phase and dynamic channel assignment phase.

5.1.1.1 Training Phase

Here, we use the traffic load and channel assignment result of AC-POCA as the training data set. Before using the data set, we need to characterize the training data into a suitable format. As we described above, we use the traffic load as the main feature to construct the input of training data. Such training data format is denoted as $tl_k = \{tl_k^1, tl_k^2, \cdots, tl_k^M\}$. Because the AC-POCA only considers the current traffic load, and the result is not affected by the traffic load of the past time sequence, we only use the traffic load of the last one slot as the input of training data.

Then, we consider that the assigned number of each link is recorded as the output of training data. If the scale of the network is significantly large, the number of links is large to make the output very complex. As mentioned in our previous work [100], the complex output will significantly decrease the training accuracy. Therefore, as the same method employed in our traffic prediction algorithm, we use $M \times E_{max}$ neural network to separately train the network, where E_{max} denotes the maximum number of active links of each node. For example, for 802.11 2.4 GHz links, because of self-interference, the same channel cannot be assigned to two links of one node. Then, the maximum number of active links E_{max} is 11, which is equal to the maximum number of channels C_{max}. Since each neural network is only used to predict the channel for one link, the number of total neural networks is equal to the number of links. And the neural network corresponding to the j^{th} link in switch j is recorded as $\{NN_{i,j} | i \leq M, j \leq E_{max}\}$. For each neural network, the output is characterized as a vector consisting of C_{max} binary elements, which can be denoted as $L = \{l_1, l_2, \cdots, l_{C_{max}-1}, l_{C_{max}} | l \in \{0, 1\}\}$. And if channel i is assigned, the

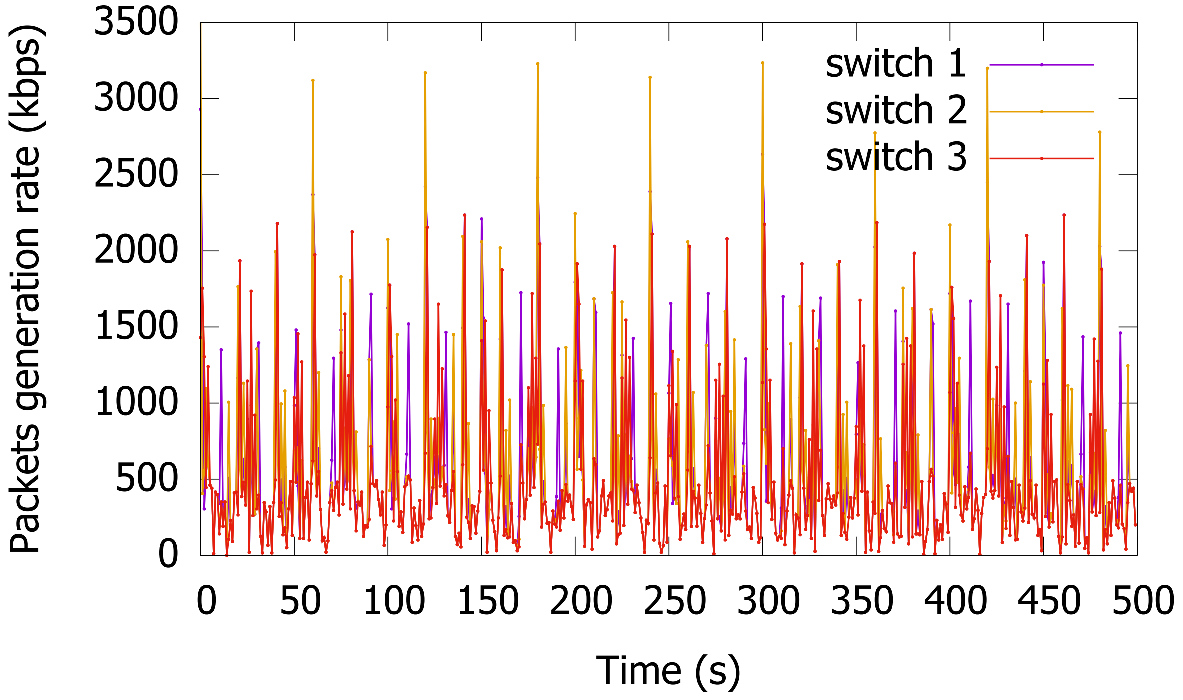


Figure 5.2: The integrated traffic pattern of different switches in the periodic intensive case (PIC).

value of the i^{th} element is 1, otherwise 0. Therefore, the training data of each neural network is indicated as $(x_{input}, y_{output}) = (tl_k, L)$.

With the training data, we try different kinds of neural network structures and different parameters for training. The comparison of the training results of different structures and parameters is shown in Sec. 5.2.2. Because of the large number of neural network and data set, this training process is better to be processed in the central controller. And the bias and weight matrices of all neural networks are recorded and updated in the central controller. The trained weight matrix of each switch is recorded as $\{WM_{i,j} | i \leq M, j \leq E_{max}\}$.

	PIC	EIC
R	100	100
Q	30	30
W	10	20
n_p	80	50
n_e	20	50

Table 5.1: The configuration of PIC and EIC

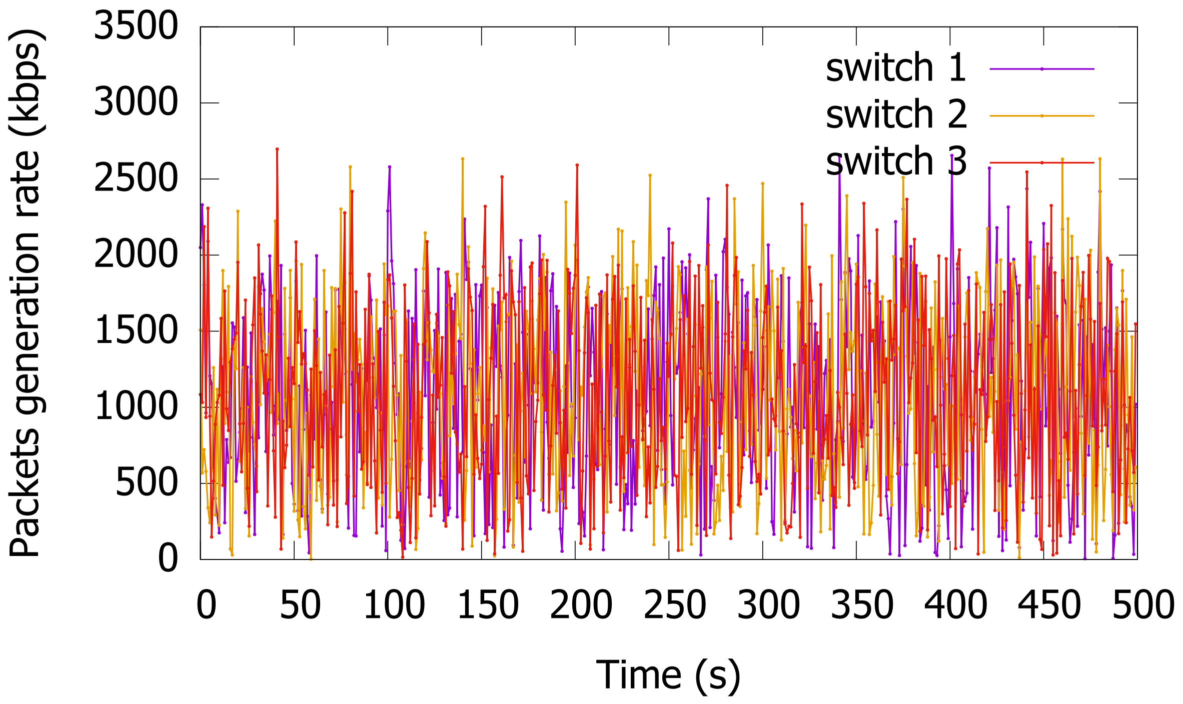


Figure 5.3: The integrated traffic pattern of different switches in the event intensive case (EIC).

5.1.1.2 Dynamic Channel Assignment Phase

After the training process, the central controller sends the copy of the trained weight matrices to each corresponding switch. Each switch only stores the weight matrices corresponding to its own links. Then, during the packet transmission period, the central controller sends the traffic load information tl_k to each switch periodically, and each switch uses the current traffic load information as the input to trigger a forward propagation process with the corresponding weight matrix to get the output L^{k+1} (i.e., the binary vector of chosen channel). If the already assigned channel L^k of the link is different from the new one L^{k+1}, the switch confirms the new channel number with the other switch on the other side of the link. If both switches get the same result, they change the channel of this link to the new one. Otherwise, the switches report the different results to the central controller. The whole process is shown in Alg. 2.

With deep learning based channel assignment, the channel assignment result can be simply obtained via a forward propagation process, which is much faster than the the game theory based channel assignment and saves most of the communication cost/signaling overhead. This is because in the conventional game theory based channel assignment methods, each router needs to keep receiving and updating the channel statements of all other routers in every iteration. Consequently, the more iterations of decision process, the heavier signaling overhead. On the other hand, in our proposed deep learning based

Algorithm 2 Algorithm of DLPOCA

```
Input: Trained weight matrices $\{WM_{i,j} | i \leq M, j \leq E_{max}\}$; each switch $\{s_i | i = 1 \rightarrow M\}$, traffic load $tl_k$.
Output: The assigned channel $L_j^{k+1}$ of each link $\{e_j | j = 1 \rightarrow E_{max}\}$.
1: for $j = 1$ to $E_{max}$ do
2:     The switch on the other side of link $e_j$ is record as $\{s_f | f \leq M\}$.
3:     $s_i$ use $tl_k$ as input to trigger forward propagation with $WM_{i,j}$, the result is recorded as $L_j^{k+1}$.
4:     if $L_j^{k+1} \neq L_j^k$ then
5:         if $L_j^{k+1} = L_f^{k+1}$ of $s_f$ then
6:             Assign $L_j^{k+1}$ to link $e_j$.
7:         else
8:             Feedback the wrong information to central controller.
9:         end if
10:    end if
11: end for
```

channel assignment, only one iteration is needed, which is the main reason why our proposal can significantly outperform the conventional one.

5.1.2 Deep Learning based Channel Assignment jointed with Prediction

Algorithm 3 Algorithm of TP-DLPOCA

```
Input: Trained weight matrices $\{WM_{i,j} | i \leq M, j \leq E_{max}\}$; each switch $\{s_i | i = 1 \rightarrow M\}$.
Output: The predicted traffic load $tlp_{k+1}$; assigned channel $L_j^{k+1}$ of each link $\{e_j | j = 1 \rightarrow E_{max}\}$.
1: if Prediction model = CTP then
2:     $s_i$ send $TL^i$ to central controller.
3:     The central controller collects all $\{TL^i | i \rightarrow M\}$, and execute CTP prediction algorithm to get $tlp_{k+1}$.
4:     Central controller sends $tlp_{k+1}$ to all switches.
5: else if Prediction model = S-CTP then
6:     $s_i$ uses $TL^i\_int^i$ as input to calculate $tlp\_int_{k+1}^i$
7:     $s_i$ sends $tlp\_int_{k+1}^i$ to central controller.
8:     The central controller collects all $\{tlp\_int_{k+1}^i | i \rightarrow M\}$ and $\{tl_k^i | i \rightarrow M\}$, then executes S-CTP prediction algorithm to get $tlp_{k+1}$.
9:     Central controller sends $tlp_{k+1}$ to all switches.
10: else if Prediction model = DTP then
11:    $s_i$ uses $TL\_rel^i$ and $TL\_int^i$ as input to calculate $tlp_{k+1}^i$ by executing DTP prediction algorithm.
12:    $s_i$ sends $tlp_{k+1}^i$ to central controller.
13:    The central controller collects all $\{tlp_{k+1}^i | i \rightarrow M\}$ to construct as $tlp_{k+1}$.
14:    Central controller sends $tlp_{k+1}$ to all switches.
15: end if
16: $tlp_{k+1}$ is used as input to execute algorithm. 2.
```

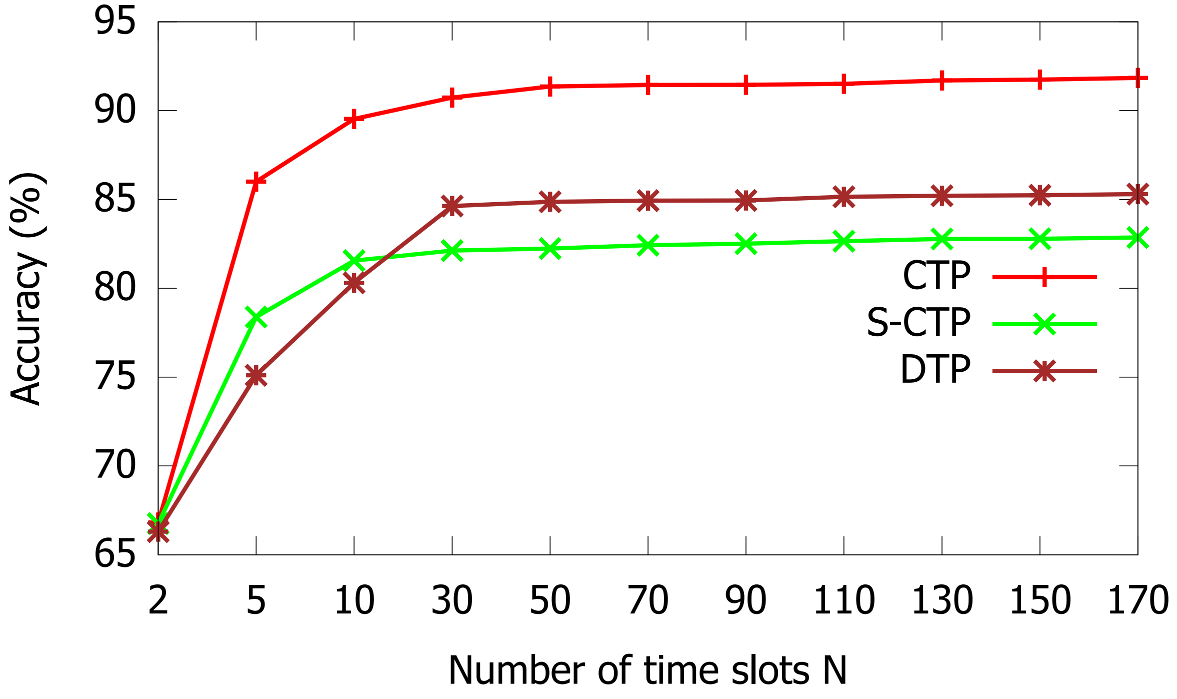
The traditional channel assignment performs the channel assignment according to the current (i.e., the last time slot) traffic load. This assumption is reasonable when the traffic load is constantly and slowly changed. However, in the practical environment, the traffic load is not so smooth. And in different applications, the traffic load may suddenly change or have more complex features. To solve this problem, we further combine the deep learning based channel assignment with the proposed traffic load prediction method, which is named as TP-DLPOCA. To compare it with DLPOCA, we replace the input traffic load in the training data with the predicted traffic load obtained via the deep learning structures described in Sec. 4. Since, the traffic load of all switches in next slot is predicted and formated as $tlp_{k+1} = \{tlp^{1}_{k+1}, tlp^{2}_{k+1}, \cdots, tlp^{M}_{k+1}\}$, the training data are denoted as $(x_{input}, y_{output}) = (tlp_{k+1}, L)$.

Except the new training data set with the predicted traffic load, the training phase of TP-DLPOCA is the same as the DLPOCA. In the dynamic channel assignment phase, the traffic load prediction is done before the channel assignment. Then, the central controller sends the predicted traffic load information tlp_{k+1} to every switch. The entired process is shown in Alg. 3.

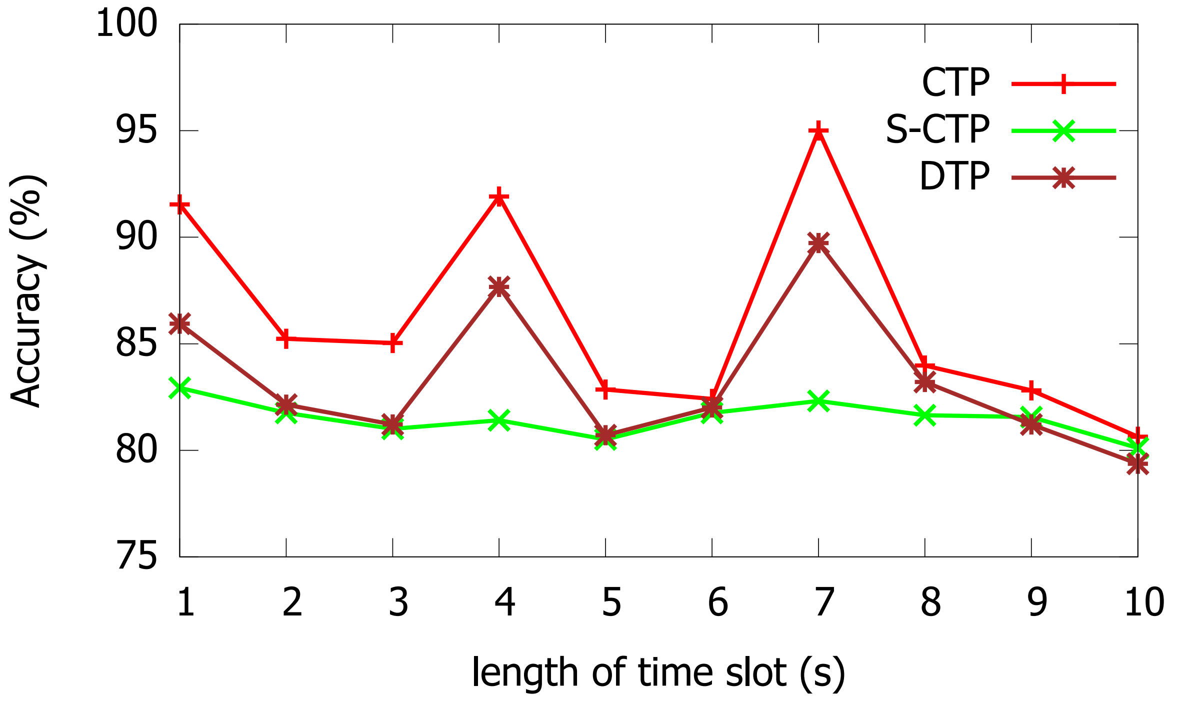
5.2 Performance Evaluation

In this section, we evaluate our proposal from three aspects: the prediction accuracy, the performance of DLPOCA, and the performance of TP-DLPOCA.

We simulate the scenarios with the configuration using C++/WILL [119] API as follow. A square area is set with same maximum width and length which is proportional to the number of switches and devices in the network. All switches in the network are randomly deployed in this area. As we described in Sec.4.0.1, there are different kinds of devices deployed in the control area of each switch. In the conducted simulations, we set the average number of devices belong to each switch area $R = 100$ and the kinds of periodic sensing devices $Q = 30$. In the beginning of simulation, we randomly choose 10 out of the Q (i.e., 30) kinds of periodic sensing devices to be deployed in the network. Consider the number of periodic sensing devices in each switch control area denoted as n_p and the number of event driven devices denoted as n_e. For simulating the influence of different kinds of devices, we considered two cases with different ratios of periodical and event driven devices deployed in the sensing plane. In one case, we deploy 80 periodic sensing devices and 20 randomly event driven sensing devices to each switch control area (i.e., $n_p = 80$ and $n_e = 20$), which represents the high ratio of periodic sensing devices, we briefly call it Periodic Intensive Case (PIC). In the other case, in each switch control area, we only deploy 50 periodic sensing devices and increase the number of randomly event driven sensing devices to 50, which represents the case with high ratio of event driven devices and briefly named as Event Intensive Case (EIC) in our research. We set the kinds of event driven sensing devices $W = 10$ in PIC and $W = 20$ in EIC. The data collection policy of event sensing devices is random. The detailed configuration and one example of integrated traffic load of different switches in the two different cases are

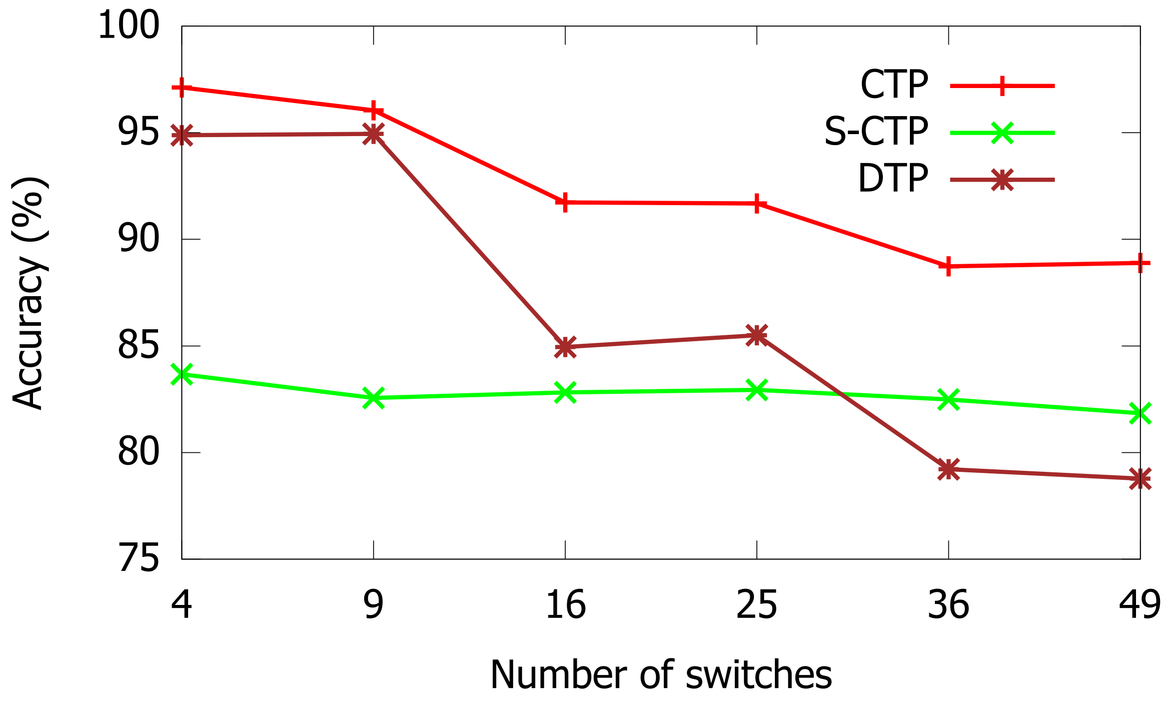


(a) The prediction accuracy with different numbers of time slots, N

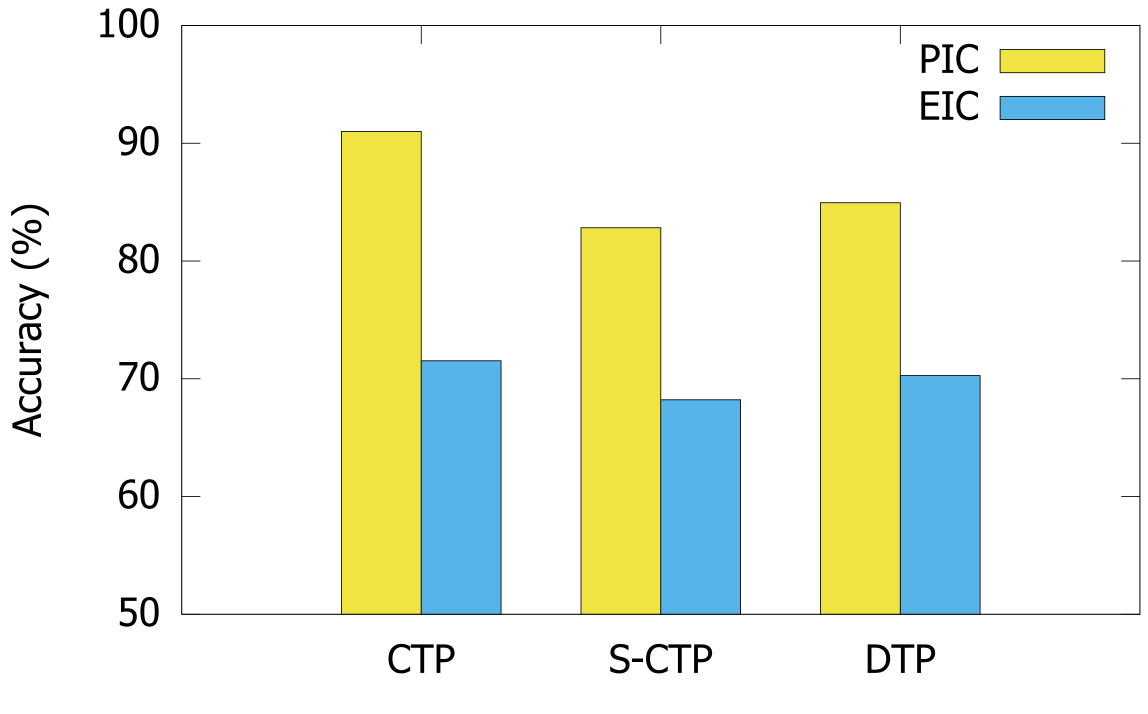


(b) The prediction accuracy with different lengths of the time slot

Figure 5.4: The performance comparison of the three kinds of proposed mechanisms with different numbers of time slots and different lengths of the time slot.



(a) The prediction accuracy with different numbers of switches



(b) The prediction accuracy with different kinds of data generation method

Figure 5.5: The performance comparison of the three kinds of proposed mechanisms with different numbers of switches and different kinds of data generation method.

shown in Table. 5.1, Fig. 6.5a and Fig. 5.3. Consider the distance between switches is randomly set ranging from 10m to the maximum width of the square place. The gateway is positioned at the top right corner in the simulated network that is the farthest from the user devices. To simplify the simulation, we use the similar spectrum configuration of [120], the multi-channels, multi-radios are assumed to be equipped on each switch and is operated with IEEE 802.11g wireless technology. The interference model of those wireless channels is mentioned in Sec. 3.2, and the $IR(\sigma)$ is shown in Table 3.1. The data rate of each link is set to 8Mbps. We conduct our network with the number of switches from 9 to 64 to show the different performances in various network environments.

5.2.1 Prediction Accuracy

At first, we evaluate the accuracy performance of traffic load prediction with three proposed mechanisms, namely, CTP, S-CTP, and DTP. In the case of PIC, the number of switches and the slot length are 16 and 1s, respectively. We compare the prediction accuracy of three mechanisms with different number of slots N, which is an important parameter of the prediction algorithm mentioned in Sec. 4.2. In Fig. 5.4a, we can notice that the accuracy of all three mechanisms increases with the increasing value of N before $N = 30$. When N exceeds 30, the accuracy slightly increases and intends to be stable. That indicate that $N = 30$ is the threshold, which represents whether the features used in the input data are enough for training. Furthermore, the figure shows that, when N is below the threshold, the accuracy of S-CTP is better than that of DTP, while the accuracy of DTP is higher when N is above the threshold. And the accuracy with CTP is always better than S-CTP and DTP (more than 90%).

With the different kinds of policies chosen by devices, the features of traffic patterns become more complex. Thus, choosing the suitable time slot Δ to fit the features of traffic pattern is very important to increase the prediction accuracy. In addition, we compare the accuracy of the three mechanisms with different lengths of Δ. This simulation is conducted in the situation of PIC, and the number of switches and the value of N are 16 and 70, respectively. The result indicates that the prediction accuracy is significantly affected by Δ. For all three mechanism, there are two crests (i.e. $\Delta = 4 \ or \ 7$) which show the most suitable slot length Δ for our traffic patterns. However, regardless of the Δ value we choose, the accuracy of CTP is always better than the other two mechanisms.

Fig. 5.5a demonstrates the accuracy of the three mechanisms with different numbers of switches. This simulation is conducted in the situation of PIC and $N = 70, \Delta = 1s$. The switches are deployed as described in Sec. 4.0.1. From the figure, we can notice that the prediction accuracies of CTP and DTP decrease with an increasing number of switches. However, S-CTP always exhibits a stable prediction accuracy. And when the number of switches is more than 25, the performance of S-CTP is even better than that of DTP. When the number of switches exceeds 25, the prediction accuracy of CTP also tends to be stable and can achieve nearly 90% accuracy. This means that the proposed deep learning based prediction algorithm is also suitable for a large scale network.

Furthermore, as shown in Fig. 5.5b, we compare the three kinds of deep learning based traffic load prediction accuracy in situations of PIC and EIC. The prediction accuracies in PIC always significantly outperforms that in EIC. This is because there are so many random events in EIC and it is hard to track its policy. On the other hand, considering the fact that the event driven still has some rules in practical networks, such situation can be further researched in the future works.

Thus, the results show the advantage of using the SDN central control system. This is because the high computation ability and communication mechanism in SDN allows more complex information to be used as training data in learning process. Next, we investigate the performance of deep learning based channel assignment in SDN-IoT.

5.2.2 Performance of Deep Learning Based Channel Assignment

In this part, we compare the learning performance of POC with different learning structures and different learning parameters. Then, we compare the POC accuracy of our proposal. Finally, we compare the throughput between our proposed DLPOCA and traditional channel assignment algorithms (i.e., the orthogonal channel assignment, POC, AC-POCA).

In Fig. 5.6, we compare the training accuracy with different learning structures, i.e., DBN with 2 and 3 hidden layers. The number of nodes in each layer is set to 20 and 100. Here, we briefly call them 20-2-DBN, 20-3-DBN, 100-2-DBN, and 100-3-DBN, respectively. Then, we change the DBN structure into deep CNN with 1 and 2 convolution layers and 2 full connection layers, respectively. In the CNN, we set the size of convolution layer as 3×3, the number of nodes in full connection layer is 100, the number of channels in convolution layer is 20, and the padding and stride are set to 1. Correspondingly, we briefly call them 1-CNN, 2-CNN. Then, we compare those different training structures in different network structures. After running all those training processes with mini-batch size of 20 and 500 epoches, the accuracy result is shown in Fig. 5.6. From the result, we can notice that the accuracy is deeply related to the training structure, and the deep CNN is much better than DBN in our scenario. Moreover, the 2-CNN can always get 100% accuracy in our network and is chosen as our final training structure.

Our proposed DLPOCA chooses the deep-CNN as the training structure, and the simulation result demonstrates almost 100% accuracy. Then, we compare the convergence time (i.e., iteration times) of DLPOCA and conventional algorithms, Cooperative Channel

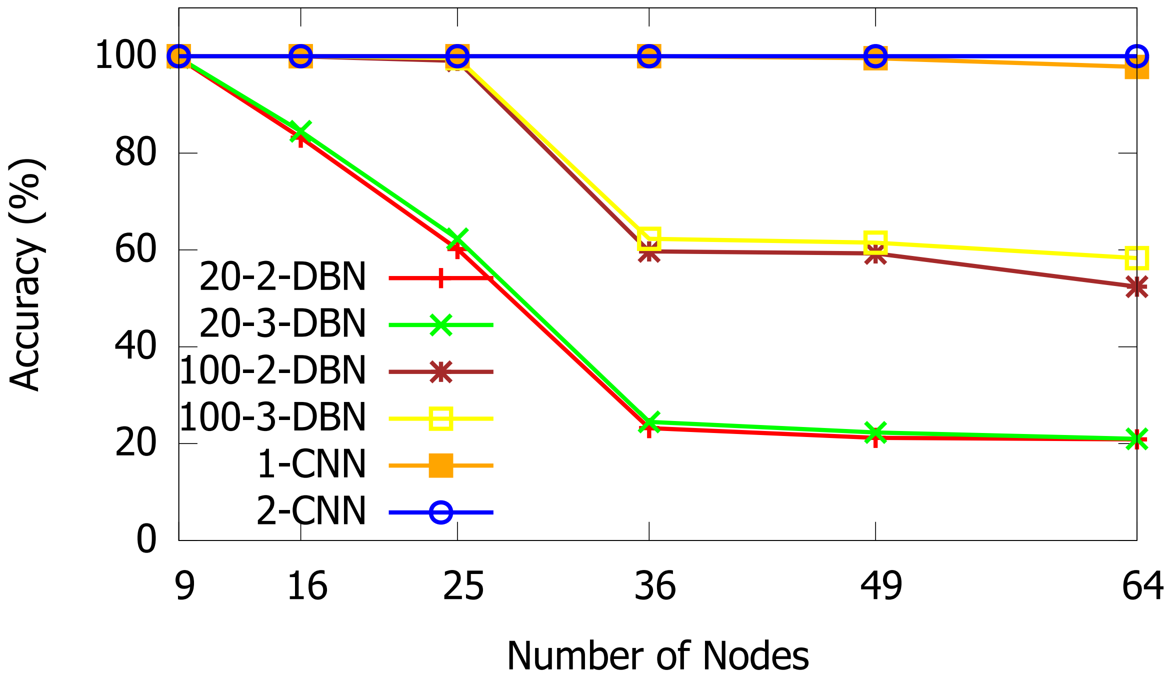


Figure 5.6: The accuracy with different configuration of learning structure.

Assignment Game (CoCAG) with Best Response (BR) and Smoothed Better Response (SBR) [29] and ACPOCA [120]. For ease of representation, the three methods are briefly referred to as CoCAG-BR, CoCAG-SBR and ACPOCA, respectively. We run the channel assignment process 100 times with randomly deployment of nodes and obtain the average number of iteration times.

Fig. 5.7 shows the comparison result of the convergence time. As described in Sec. 5.1.1, with our proposed DLPOCA, the number of iteration times is always 1, which significantly outperforms conventional algorithms. In conventional algorithms, the switch chooses the channel of its links depends on the decisions of other switches. This means that the switches must wait until other prior switches finished their channel assignment. The more the iteration times, the longer time each switch needs to spend in channel assignment. This causes redundant convergence time.

Because of the redundant convergence time, redundant signaling correspondingly increases. During the convergence time, all links are down because of the channel reassignment, and the throughput decreases with such redundant convergence time. However, with our proposed DLPOCA, the number of iteration time is always 1, and both the convergence time and signaling overhead are significantly low.

Considering the dynamics of network, the traffic load and link condition may change frequently, leading to the frequent channel reassignments. Here, we consider the traffic load in the situation of PIC. In order to simulate such a situation, we set the frequency

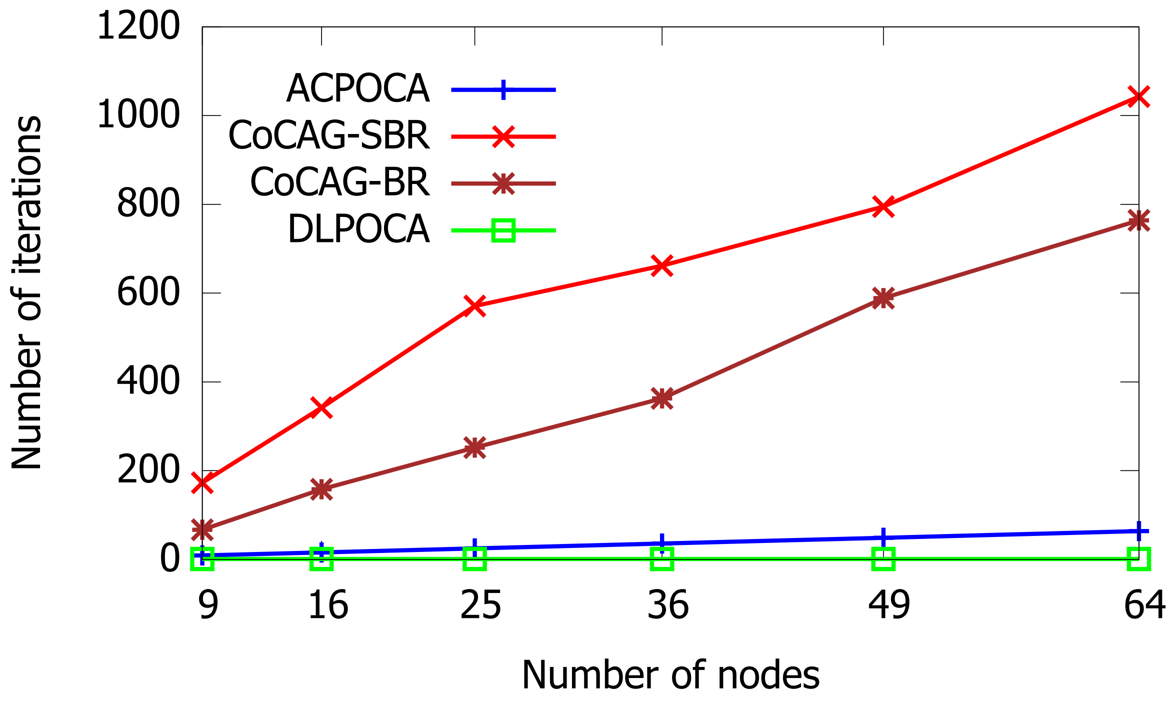


Figure 5.7: The convergence compared with proposal and conventional algorithm.

of channel assignment in our simulation as 10s/1 (i.e., every 10 seconds execute once). During the channel assignment, all data transmissions are paused. And then, we compare the throughput between DLPOCA, ACPOCA and CoCAG. In this simulation, the buffer of each node is set to 100KB, and the packets are randomly discarded when the buffer is full. In order to show the advantage of our proposal more clearly, in this section, we just consider that all packets are generated normally, i.e., not in the situation of PIC or EIC. The packet generation rate of each node is set as 1Mbps. The packet size and signaling size both are set to 1kb. Then, we run the simulation over 1000s, and the throughput result is demenstrated in Fig. 5.8. From the above result, it can be noticed that the throughput of the proposed DL-POCA is always better than that of conventional channel assignment algorithms. This is because of the quick convergence of our proposal. The channel assignment of all routers can be decided only with one broadcast and finished almost immediately. On the contrary, in the conventional channel assignment algorithms, the switches need to make decisions one by one that causes a high number of iterations to converge. Especially by randomly Smoothed Better Response based game theoretical CoCAG, switch have to make many redundant decisions to maximize their utility.

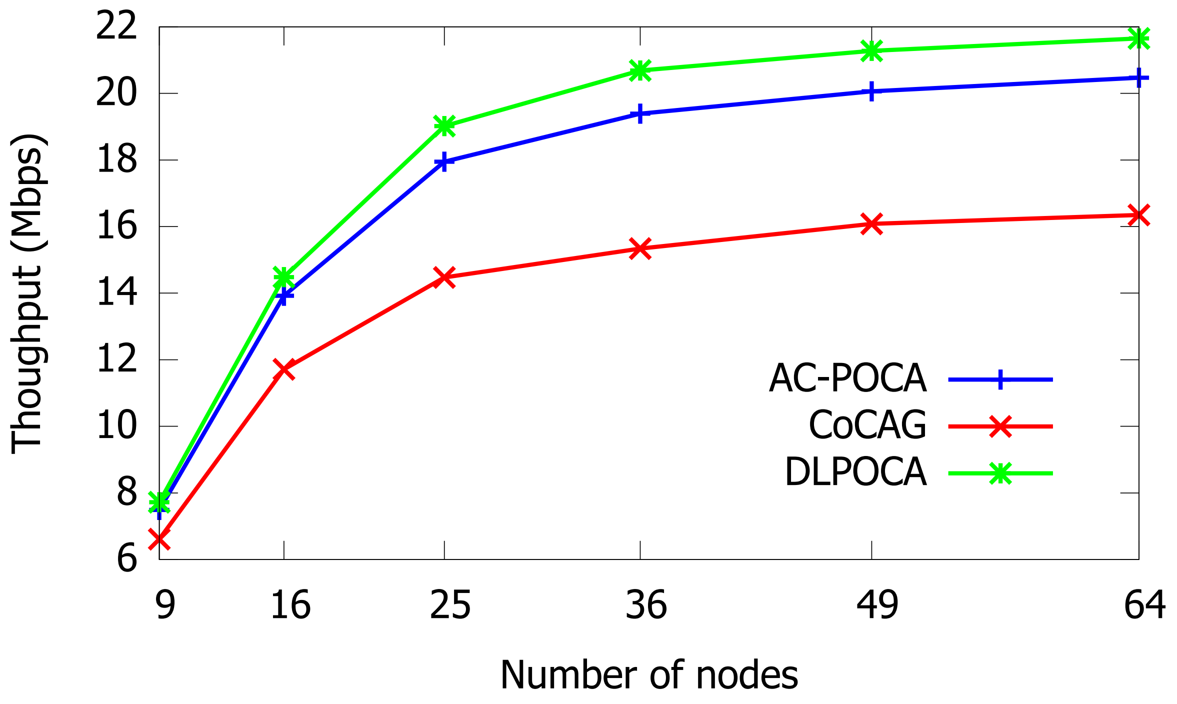
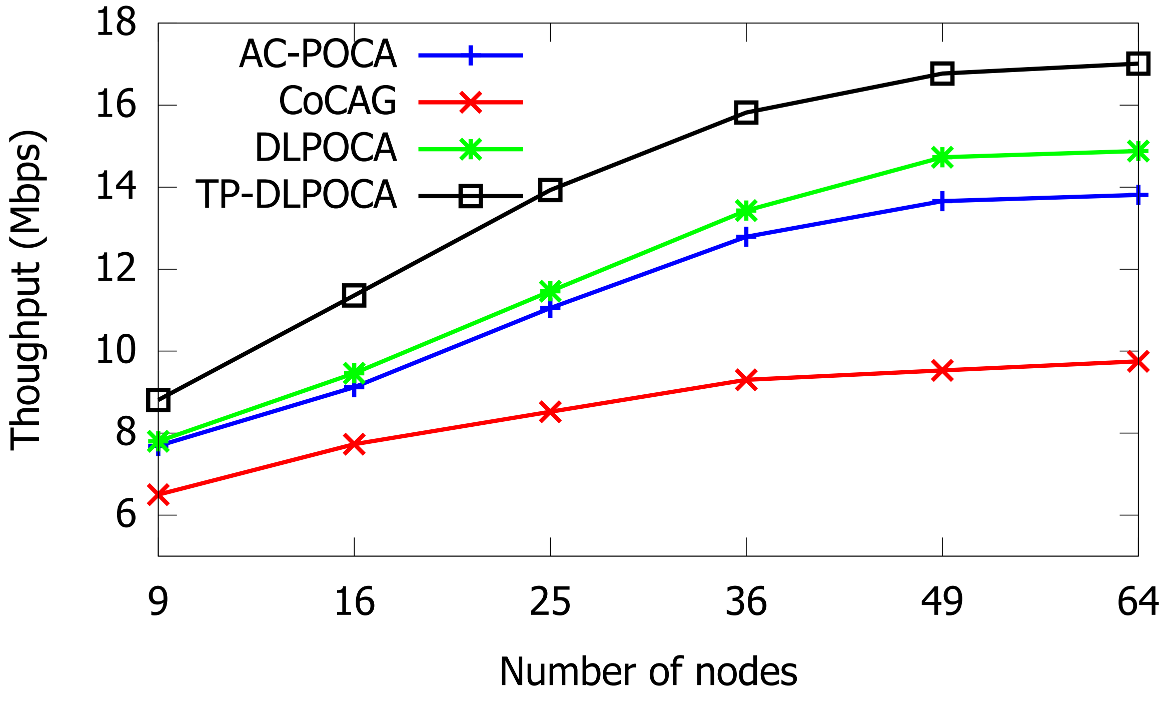


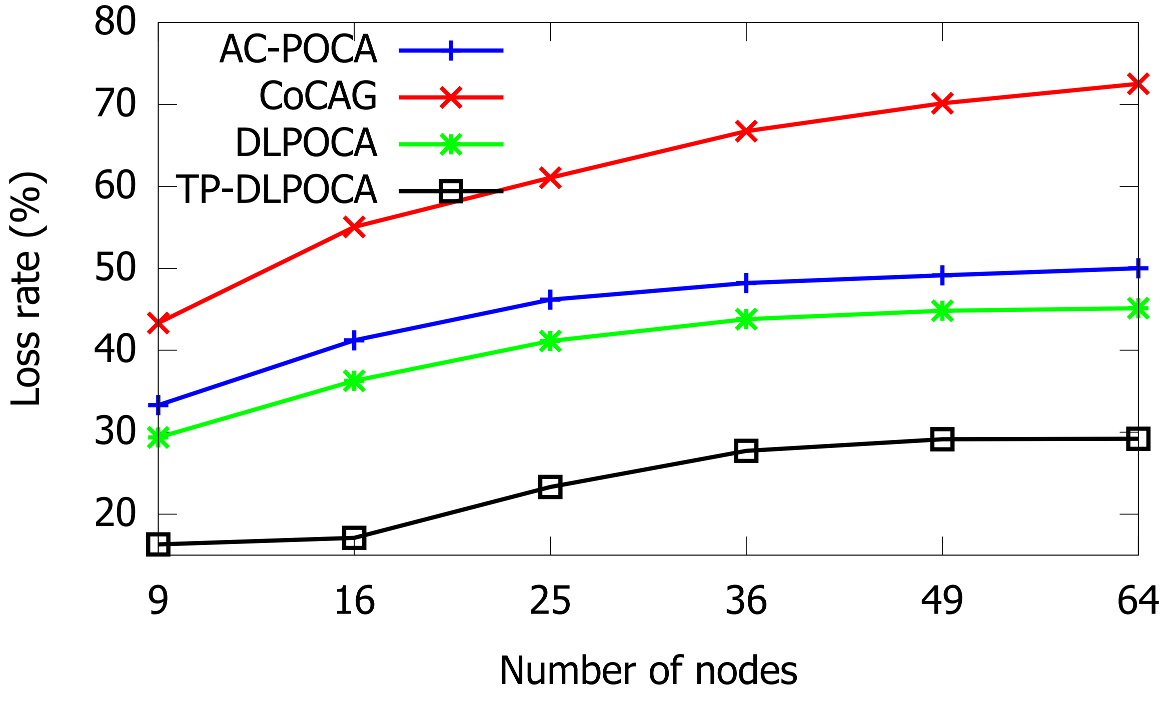
Figure 5.8: The throughput compared with proposed DLPOCA and conventional algorithms.

5.2.3 Performance of the Joint Deep Learning Based Prediction and Channel Assignment

To further improve the channel performance, we combine the intelligent deep learning based channel assignment with deep learning based traffic load prediction. The result of Fig. 5.9 shows the performance of our proposed TP-DLPOCA with conventional channel assignment algorithms (i.e., CoCAG, AC-POCA). In this simulation, the parameters are considered to be the same as those in Sec. 5.2.2. Figs. 5.9a and 6.4 demonstrate the comparison of throughput and packet loss rate of TP-DLPOCA and conventional algorithms, respectively. From the results, we can notice that the performance of the proposed TP-DLPOCA is much better than that of conventional algorithms and even DLPOCA. Furthermore, when the number of switches increases, the packets loss rate of TP-DLPOCA almost tends to be stable, indicating the advantage of TP-DLPOCA compared with conventional algorithms in a larger network. This is because TP-DLPOCA can predict the congestion and give the links in the congestion area higher priority to be assigned with high quality channels. And even when the network situation changes, the proposed TP-DLPOCA can still learn from the new situation and predict the congestion to assign suitable channels. This good performance of the proposed TP-DLPOCA can be credited as the online intelligent channel assignment strategy.



(a) The throughput compared with TP-DLPOCA and conventional algorithms



(b) The packets loss rate compared with TP-DLPOCA and conventional algorithms

Figure 5.9: The network performance comparison of the TP-DLPOCA, DLPOCA and conventional algorithms in terms of throughput, packets loss rate in the situation of PIC.

5.3 Summary

The explosive growth of sensing data and quick response requirements of the IoT have recently led to the high speed transmissions in the wireless IoT to emerge as a critical issue. Assigning suitable channels in wireless IoT is a basic guarantee of high speed transmission. However, the conventional fixed channel assignment algorithms are not suitable in the IoT due to the highly dynamic traffic loads. Recently, the Software Defined Networking based IoT (SDN-IoT) is proposed to improve the transmission quality. Moreover, the deep learning technique has been widely researched in high computational SDN. Therefore, a deep learning based partially channel assignment algorithm (DLPOCA) was proposed to intelligently assign channels to each link in SDN-IoT. Then, by using the previous proposed traffic load prediction method to predict the future traffic load and network congestion, we combine the traffic prediction and channel assignment to propose a novel intelligent channel assignment algorithm (TP-DLPOCA), which can intelligently avoid traffic congestion and quickly assign suitable channels to the wireless links of SDN-IoT. Extensive simulation results demonstrate that our proposal significantly outperforms the conventional channel assignment algorithms.

Chapter 6

Proposed Deep Learning Based Network Traffic Allocation

As mentioned in the introduction, in my thesis, I mainly consider the radio resource allocation and traffic resource allocation problem in the IoT. In previous works, we solved the radio resources allocation problem in IoT. However, in the considered SDN-IoT, especially in the Data plane, how to allocation the integrated traffic flows from sensing plane then becomes the main concern in the SDN-IoT. In this chapter, we consider the traffic allocation problem in the data plane of SDN-IoT which can be treated as a SDN enabled backbone in IoT. In conventional network, the network traffic control especially the routing protocol are widely researched to offer the traffic allocation service in network. However, the networks still operate on routing frameworks that were designed decades earlier. Indeed, as the wireless networks continue to evolve, efficient network traffic control such as routing methodology in the wireless backbone network appears as a key challenge [121]. The existing routing protocols used in such networks were designed originally for the fixed, wired networks that rely on calculating the shortest path from a source to its destination based on distance vectors or link costs [122, 123, 124, 125]. To conquer the challenges, in this chapter, I further introduce a deep learning based network traffic allocation (i.e., network traffic control) mechanism.

The application of deep learning for network traffic allocation, in wireless/heterogeneous networks is a relatively new area. With the evolution of wireless networks, efficient network traffic control such as routing methodology in the network appears as a key challenge. This is because of the reason that, firstly, the conventional routing protocol requires lots of global information from all other nodes which leads to high signaling overhead, in second, the conventional routing protocols do not learn from their previous experiences regarding network abnormalities such as congestion and so forth. Therefore, in the future high speed network, an intelligent network traffic control method is essential to avoid those problems.

As mentioned in the related work part, my previous works [100, 103] mainly used supervised learning to solve the signaling problem of traffic allocation. In this chapter, I address the second issue and propose a new, real-time deep learning based intelligent network traffic allocation method. The proposed method do not depends on the existing labeled data and is based on online self-learning which exploiting deep Convolutional Neural Networks (deep CNNs) with uniquely characterized inputs and outputs to represent the considered backbone in SDN-IoT. Simulation results demonstrate that our proposal achieves significantly lower average delay and packet loss rate compared to those observed with the existing routing methods. We particularly stress on our proposed method's independence of existing routing protocols that make it a potential candidate to remove routing protocol(s) from future wired/wireless networks especially in the considered SDN-IoT scenario.

6.1 Problem Statement and Considered Deep Learning System

In this section, we first formulate the problem statement, and then present our considered deep learning system model.

To describe our research problem in an easy manner, we consider a the backbone topology of the data plane in the SDN-IoT consisting of several SDN switches to serve the IoT devices as depicted in Fig. 6.1. The mechanism to choose one route from a number of alternative paths to connect each source-destination pair in such a communication network is referred to as a routing strategy. Let N denote the number of existing switches (i.e., routers) in the network that are represented by the set, $\mathbf{R} = \{r_0, r_1, \ldots, r_{N-1}\}$. The routing strategy in a network can be formulated as a classical combinatorial optimization problem, i.e., the shortest path routing problem in a graph. However, conventional routing protocols (such as OSPF, Intermediate System to Intermediate System (IS-IS), Routing Information Protocol (RIP), and so forth) are inherently prone to the same problem when the network environment degrades. In particular, when the network becomes heavily congested, conventional routing protocols typically make routing decision, which is retained even if the same/similar congestion events recur at a later instant. This is because the conventional routing protocols, designed decades earlier, adhere to making decisions based on fixed rules or policies. In other words, the traditional routing strategies are not intelligent and therefore, they repeatedly make the same routing decision for similar congestion scenarios triggered by the burst traffic that suddenly changes during a short time interval. For example, as depicted in Fig. 6.1, the source switch r_0, r_3 and r_6 receive integrated input packets from sensing plane and send them to the destination

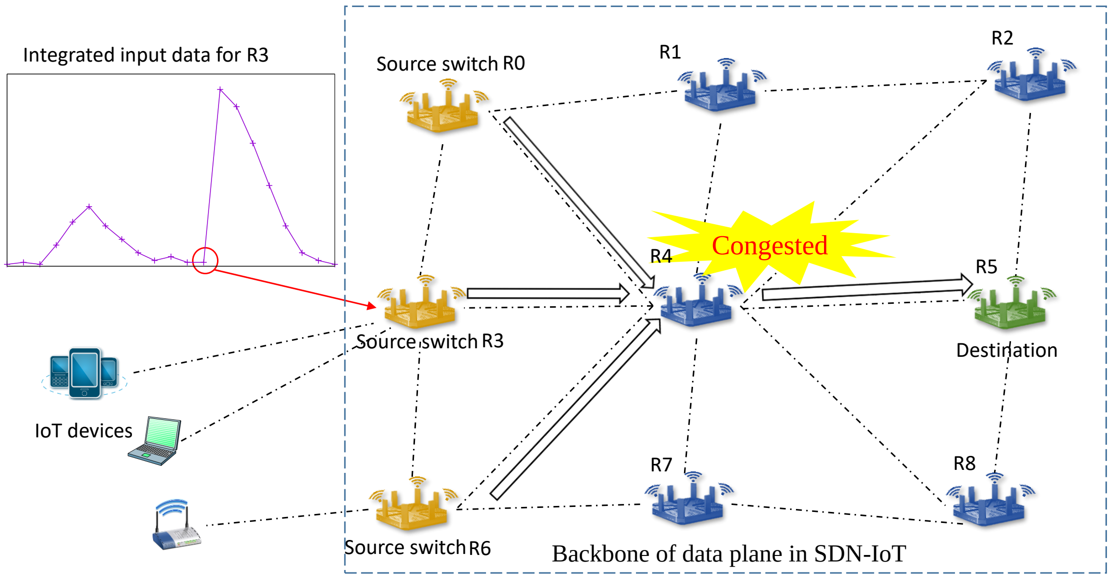


Figure 6.1: Considered wireless network backbone and depicting our focused problem.

switch r_5. Prior to the appearance of burst traffic at the source switch, the load of switch r_4 is small, and hence, the traditional routing method chooses r_4 to forward the packets to r_5. However, when the source switch suddenly experience the burst traffic, the load of r_4 increases dramatically which leads to congestion at r_4. In order to deal with such a network congestion, the packets are forwarded via alternative paths (e.g., through r_1 and/or r_7) to relieve the burden at r_4. However, when such a situation recurs, the conventional routing method always makes the same decision to combat the same/similar congestion event. This is because the traditional protocol is "non-intelligent", i.e., not able to learn from the past events and "remember" how to deal with such scenarios.

In order to intelligently make routing decisions in a the considered network as shown in Fig. 6.1, in this chapter, we adopt a deep learning system. Our considered deep learning system model is presented in the remainder of the section. The deep learning system can collect past "errors" (i.e., ineffective routing decisions) and their corresponding events (i.e., congestion, network performance degradation, and so forth) to predict and avoid the same errors when similar situations recur. In other words, when a certain routing strategy and input traffic pattern are given, by employing the trained deep learning system, a certain output can be obtained so as to indicate whether this routing strategy may cause congestion or not. As our purpose is to learn the identifiable features from the network traffic, we employ a strong feature learning system, namely deep CNN introduced in Chapter.4, to construct our learning system as depicted in Fig. 6.3a. The deep CNN comprises two main components, namely the feature extraction and classification parts. The training process is similar to the pre-mentioned Deep-CNN in the traffic load prediction part. In the feature extraction part, many convolution layers are used to filter the low level features of the input data while the pooling layers are used to progressively reduce the size of features and parameters, and improve computation in the network. The convolution and pooling layers are employed to eventually extract features of the input data. Based on those extracted features, the classification part carries out the final training process. This part is a little different from the previous prediction process, the fully connected layers provide the core workspace to compute the extracted input data and outputs as an N-dimensional vector. Different from the continuous value of predict traffic load in prediction part, the output in this process are constructed as discrete value denotes the result of the final classification.

Next, assuming that a routing strategy is given, we set the traffic patterns of each node (i.e., switch) and the state of congestion in the network as input and output. Consider the traffic pattern of the network consists of different kinds of information (features), such as packets generation rate, waiting queue length in buffer, and so forth. To better utilize the powerful matrix computing ability of deep CNN, we characterize the input data format into a 3-dimensional matrix, $(\mathbf{CN}, \mathbf{T}, \mathbf{R})$, which is similar to the data format shown in Fig. 6.3b where $\mathbf{CN} = \{cn_0, cn_1, \cdots, cn_{M-1}\}$, and M denotes the number of data channels (Different from the wireless channel in network, the "'channel"' is a concept used in CNN to denotes the different features extracted from the input) used as input. If we only consider a single channel cn_j, the input data of each channel is recorded as a 2-dimensional matrix $(\mathbf{T}, \mathbf{R})$. Because the traffic pattern is constructed by a time series containing many time intervals, only a single time interval is not sufficient to describe all the features of traffic patterns. Therefore, we use a number of time slots to record adequate features. $\mathbf{T} = \{t_\beta, t_{\beta-1}, \cdots, t_{\beta-H+1}\}$, where t_β means the current time interval, and H indicates the number of time intervals used as an input sequence. $\mathbf{R} = \{r_0, r_1, \cdots, r_{N-1}\}$, as mentioned earlier, means different nodes (i.e., routers). Thus, as shown in the matrix in Fig. 6.3b, the different rows record the features of different time intervals while different columns record the features of different switches. Furthermore, different platforms indicate records in different channels. For example, consider the generation rate recorded as the second channel. Then, the generation rate of switch r_5 during last time interval is recorded in column 5, line 1 of channel 2. Corresponding to the input matrix, as mentioned above, our output is simply characterized as a 2-dimensional vector, and each of its element has a binary value. For example, we can set $(1, 0)$ as the notation of congestion.

All the weights in the deep CNN are initialized with a random function, i.e., Gaussian function and Xavier function. With the input and output, the assigned deep CNN is running to train the neural network to reduce the error until a reasonable weight matrix of the neural network is obtained. Then, such a trained neural network (i.e., its weight matrix) can be used to provide the desired output when a certain input traffic pattern is given.

6.2 Proposed Deep Learning Based Network Traffic Control Method

In this section, we describe our proposed deep learning based intelligent network traffic control method. It is an online algorithm, which can be used in any existing routing protocol to improve them or used to evolve into a new routing strategy. In other words, our proposal aims to be independent of any baseline routing protocol. In our proposal, the entire process can be divided into four steps, i.e., initial, running, updating, and training phases. These steps are shown in the flow chart of Fig. 6.2 and described in the remainder of the section.

6.2.1 Initial Phase

Prior to the running phase, each switch needs to complete several tasks as follows. First, each switch r_f calculates all possible paths, $p_{f,i,j}$, to each destination node r_i. The path number j is recorded as a 2-dimensional matrix. Additionally, $\{p_{f,i,j} \in P_f | f \leq N, i \leq m, j \leq n\}$, where m $(\leq N)$ and n denote the number of destinations and the maximum number of possible paths to each destination, respectively. P_f indicates the set of all possible paths combination to all destination routers of r_f and is recorded as a 3-dimensional path matrix. In the path set P_f, all $p_{f,i,j}$ paths are arranged as a minimum priority queue depending on the metric value (e.g. hop number, distance) of each path. For example, in switch r_1, the third order path to destination r_5 is expressed as $p_{1,5,3}$. We assume that there is a switch r_{sup} with sufficient computation ability that collects P_f of all switches and records as a 4-dimensional matrix $P = \{P_1, P_2, \cdots, P_{N-1}, P_N\}$. This is done for the purpose of avoiding congested paths and quickly choosing the valid alternative path in the minimum priority queue of each router. In addition to the total paths combination containing all routing data to each destination, another routing strategy combination set C is employed to only record the paths number combination c of all routers as an N-dimensional vector $\{c_{(j_1,j_2,\cdots,j_{N-1},j_N)} | j <= n, c \in C\}$.

6.2.2 Running Phase

In the running phase, we have two periods, namely the cold start period and intelligent running period. In the cold start period, before the training phase learns enough situations and routing judgment of updating phase can be done, we choose the minimum hop paths combination P^{short} (i.e., the first order path in the tensor) of all routers from the paths set P as the first routing paths combination. Then, after r_{sup} collects enough training data and the training phase is accomplished, the running phase goes into intelligent running period. In the intelligent running period, the real-time updating phase is

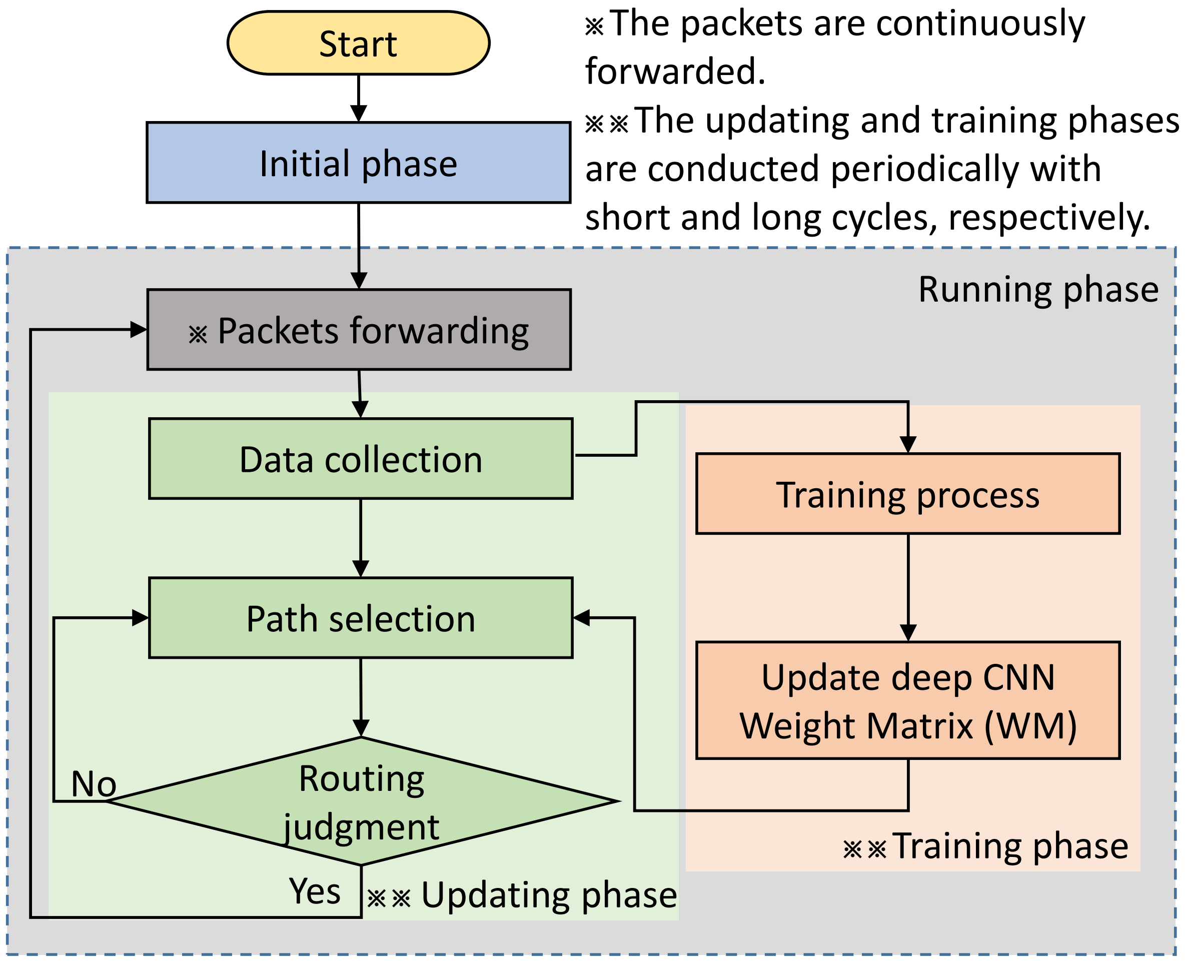

Start
Initial phase
※The packets are continuously forwarded.
※※The updating and training phases are conducted periodically with short and long cycles, respectively.
Running phase
※ Packets forwarding
Data collection
Training process
Path selection
Update deep CNN Weight Matrix (WM)
No
Routing judgment
Yes
※※ Updating phase
※※Training phase

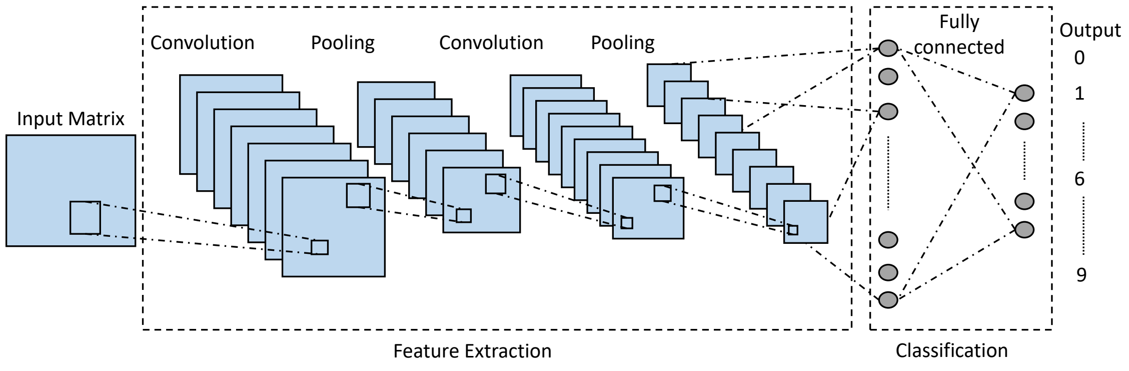
Figure 6.2: The flow chart of our proposed deep learning based network traffic control method.

processed periodically. Furthermore, the valid paths combination P^{valid} are intelligently chosen and executed by the result of the routing judgment in the updating phase.

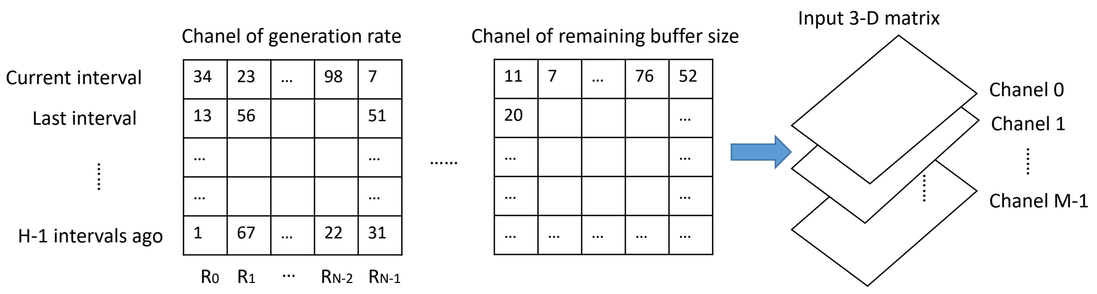
6.2.2.1 Real-time Updating Phase

The updating phase collects traffic patterns for training data and performs routing judgment to intelligently choose the valid paths combination and find the best path to avoid congestion. During the updating phase, there are two steps, namely, data collection, and path selection and routing judgment.

- Data Collection: The routing information and traffic patterns should be collected and updated to r_{sup} in a certain time period, t_{up}. When the packets arrive at the destination switch, the switch records the transmission delay t_d of each packet. During the updating phase, r_{sup} records the routing strategy combination c. In addition, each switch calculates the average packets delay and packets loss rate



(a) Considered deep CNN structure.



(b) Input characterization.

Figure 6.3: Our unique input characterization for the deep CNN.

during the period between the last and the current updating phases, and sends those information to r_{sup}. Here, we refer to the period between two consecutive updating phases as the updating interval, t_{up}. Then, a threshold th is preset (by calculating from the maximum and minimum values of all periods) to judge whether the network is congested or not during this t_{up}. The result is recorded as $\{y^{(2)}|(1,0)(0,1)\}$. The occurrence of the congestion is recorded as $(1,0)$. Otherwise, $(0,1)$ is recorded. Also, the traffic patterns including packets generation rate and packets waiting queue size in the buffer of each router are recorded as $\{x^{(i)}|i = 1, \ldots, N\}$. To improve the accuracy, the traffic patterns are always recorded over h intervals where $(h > 1)$. Both the congestion state and traffic patterns are finally combined as a labeled training set, $\{D = (x^{(i \times h)}, y^{(2)})|i = 1, \ldots, N; h > 1,\}$. After several updating phases, the training set becomes large enough to train the deep CNN, i.e., to start the training phase.

- Path Selection and Routing Judgment: In r_{sup}, for each routing strategy c, there is a corresponding deep neural network's WM to judge whether the recent traffic patterns will cause congestion or not. If the result is indeed, a congestion, we denote this paths combination, c as invalid. The switches in the network choose the next combination, c_{next}, and perform the above judgment again until a valid paths combination is chosen. For example, when $c_{(1,\cdots,1)}$ is assessed by $WM_{1,\cdots,1}$ as invalid, c is changed to $c_{(2,\cdots,1)}$ and judged by $WM_{2,\cdots,1}$ again. Until $c_{(2,3,\cdots,5)}$ is evaluated by $WM_{2,3,\cdots,5}$ to be valid, the network will choose the corresponding paths combination p of strategy combination $c_{(2,3,\cdots,5)}$ as a valid routing path p^{valid} during the next updating interval. Next, we describe how the neural network's WM is constructed and improved through the training phase.

Table 6.1: Considered features of the deep CNN.

Input layer		Convolutional layers			Full connection layers			Output layer	
Width	3	layer	1	2	layer	1	2	node	2
		width	3	3					
		height	3	3	node	100	15		
height	10	channel	20	30				active	softmax
		stride	1	1	active	relu	relu		
		padding width	1	1					
channel	2	padding height	1	1				initialize	xavier
		active	relu	relu	initialize	xavier	xavier		
		initialize	xavier	xavier					

6.2.2.2 Online Training phase

The training phase learns the congestion state of each combination c to train the network and improve the routing judgment accuracy of the updating phase. After the training data D is collected in the updating phase, the deep CNN is created according to the paths combination C. For example, if the network route is in $c_{(1,3,\ldots,3)}$, then the neural network $WM_{1,3,\ldots,3}$ is constructed for training. If $WM_{1,3,\ldots,3}$ already exists, then the training process begins to improve the deep CNN network as mentioned in Sec. 6.1. These trained neural networks (i.e., their weight matrices) are used in the updating phase to make the routing decision. With such an online (i.e., real-time) training algorithm, the neural networks can be trained periodically with the training interval, t_{in}. It is worth noting that the training interval t_{in} is set as a large number of updating intervals (i.e., t_{up}s) so as to obtain sufficient labeled training data. Then, the neural network WM can be improved over time so as to make the routing decision with higher accuracy.

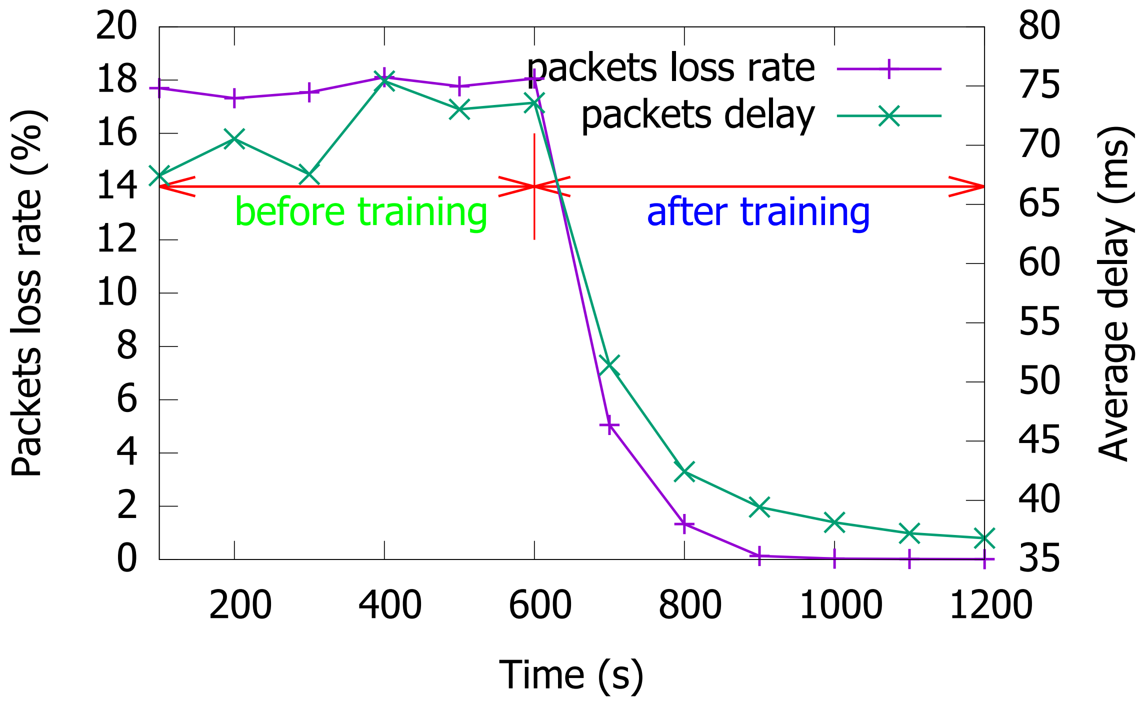


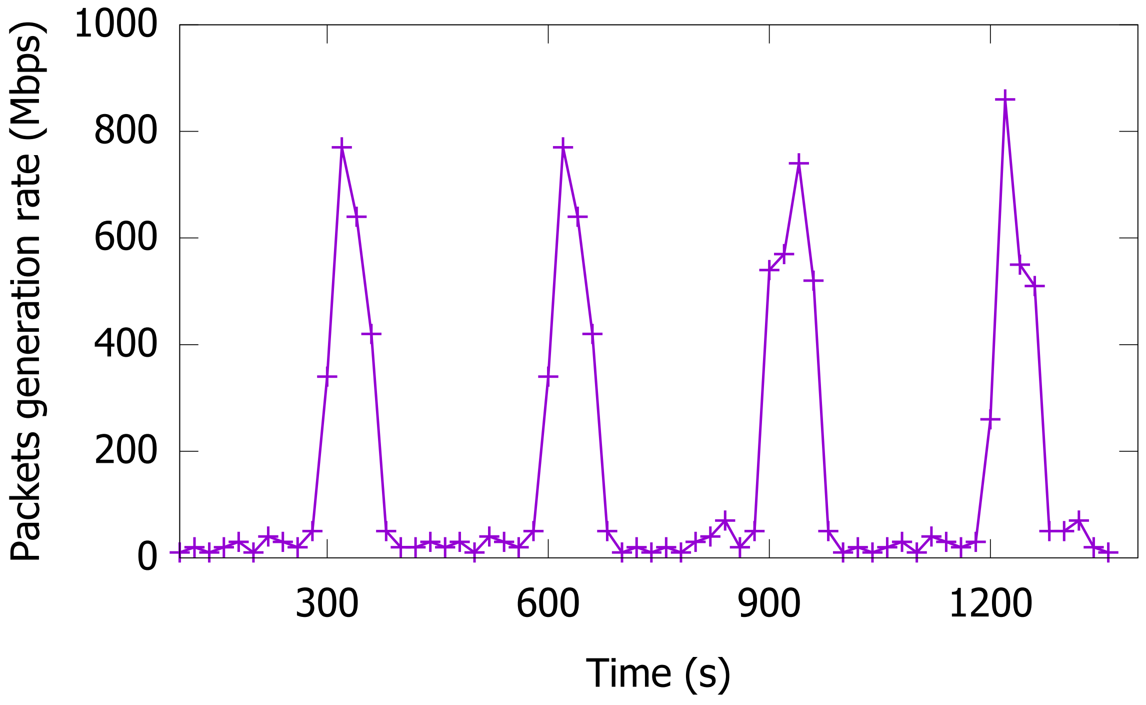
Figure 6.4: Packet loss rate before and after the training phase of proposal.

6.3 Performance Evaluation

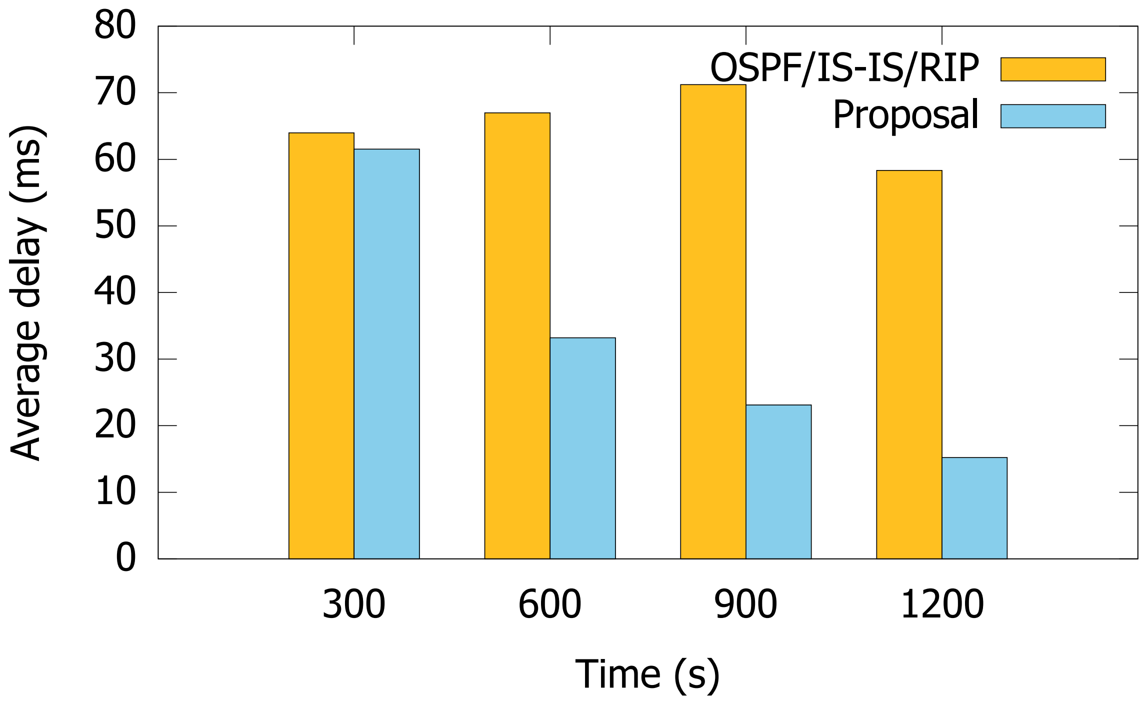
In this section, we evaluate the performance of our proposed deep learning based traffic control method by conducting simulations using C++/WILL [119]. We compare the performance of our proposal and conventional routing algorithm including OSPF, IS-IS, and RIP. For simplifying our conducted simulations, we consider our scenario as a 3x3 SDN backbone with 9 switches in the data plane of SDN-IoT, topology of which is depicted in Fig. 6.1. In such a network, we consider each node as a switch. The simulation is conducted on a workstation with Intel Core i7-6900K CPU, 64GB Random Access Memory (RAM), and Nvidia Geforce TitanX GPU. Here, we consider r_0, r_3, r_6 as source switches which can generate integrated packets from sensing plane and send them to the destination r_5, and all switches are assumed to receive and forward packets. The link bandwidth between each switch is set to 800Mbps. The buffer of each switch is set as 10MB, and the packets are randomly discarded when the buffer is full. At first, we set the average packet generation rate of each source switch as 180Mbps, and the data generation process is set as a Poisson process (with means=2). Consider at the initial phase, all switches choose the shortest path to send packets. Then, our proposal is used to train the new routing paths. We set the data updating interval (t_{up}) and training interval (t_{in}) as 1s and 100s, respectively. The layers of the deep CNN are set to two convolution layers and two fully connected layers. Because of the manageable size of the input data, for the conducted simulation, no pooling layer is considered. The specific features of the deep neural network are shown in Table 6.1. The total running time is 1200s, the data collection phase begins since the beginning, and the training phase commences at 600s.

From Fig. 6.4, we can see that, without the intelligent training process, the switches only choose the first paths combination, the packets delay and packet loss rate always remain in a high level, which indicates congestion. When the training begins, both packet loss rate and packets delay significantly decrease. This is because the network learned the solution to avoid the congestion and chooses valid paths combination by training the labeled routing data. After running 500s of the training phase, the number of congestion occurrences becomes almost zero. This shows a significant performance improvement in avoiding congestion using our proposed real-time learning exploiting deep CNNs. From Fig. 6.4, we can see that, without the intelligent training process, the switches only choose the first paths combination, the packets delay and packet loss rate always remain in a high level, which indicates congestion. When the training begins, both packet loss rate and packets delay significantly decrease. This is because the network learned the solution to avoid the congestion and chooses valid paths combination by training the labeled routing data. After running 500s of the training phase, the number of congestion occurrences becomes almost zero. This shows a significant performance improvement in avoiding congestion using our proposed real-time learning exploiting deep CNNs.

Next, we compare the performance of our proposal with existing routing algorithms, i.e, RIP, IS-IS, and OSPF. In order to simulate a network congestion, we use different kinds of bursty input data in each different source node. One of them is shown in Fig. 6.5a for generating the packets at one of the source nodes. Note that the average packets generation rate is set to 180Mbps. Fig. 6.5b demonstrates the comparative performances. As RIP, IS-IS and OSPF employ similar graph-based algorithms for determining the routing paths, they exhibit similar performances. From the result, we can notice that with ongoing time, the delay and packet loss rate always stay in a high level with the traditional routing protocols. However, they decrease significantly with our proposal. Such trends indicate that the traditional routing algorithms always deal with the sudden traffic surge instances in a similar manner, and hence, result in network congestion. In other words, because of the lack of learning process, when a similar input traffic appears, the conventional routing methods always choose a similar strategy and repeatedly confront congestion. Compared with those traditional algorithms, when a congestion happens, our proposal is able to learn the congestion state so that it may avoid the same problem at a later instant. With the same learning features as shown in Table 6.1, our proposal running for more than 1000s (i.e., 10 learning phases) achieves a significantly high learning accuracy (i.e., up to 98.7%). This means that our proposed intelligent routing decision faciliated by the deep learning methodology can avoid almost 98.7% congestion cases. Thus, our proposal allows real-time learning to achieve better performance with time.



(a) Input traffic exhibiting bursty nature used in the conducted simulation.



(b) Average delay comparison during the course of simulation.

Figure 6.5: The input traffic and performance comparison of our proposal and conventional OSPF, IS-IS, and RIP of average delay

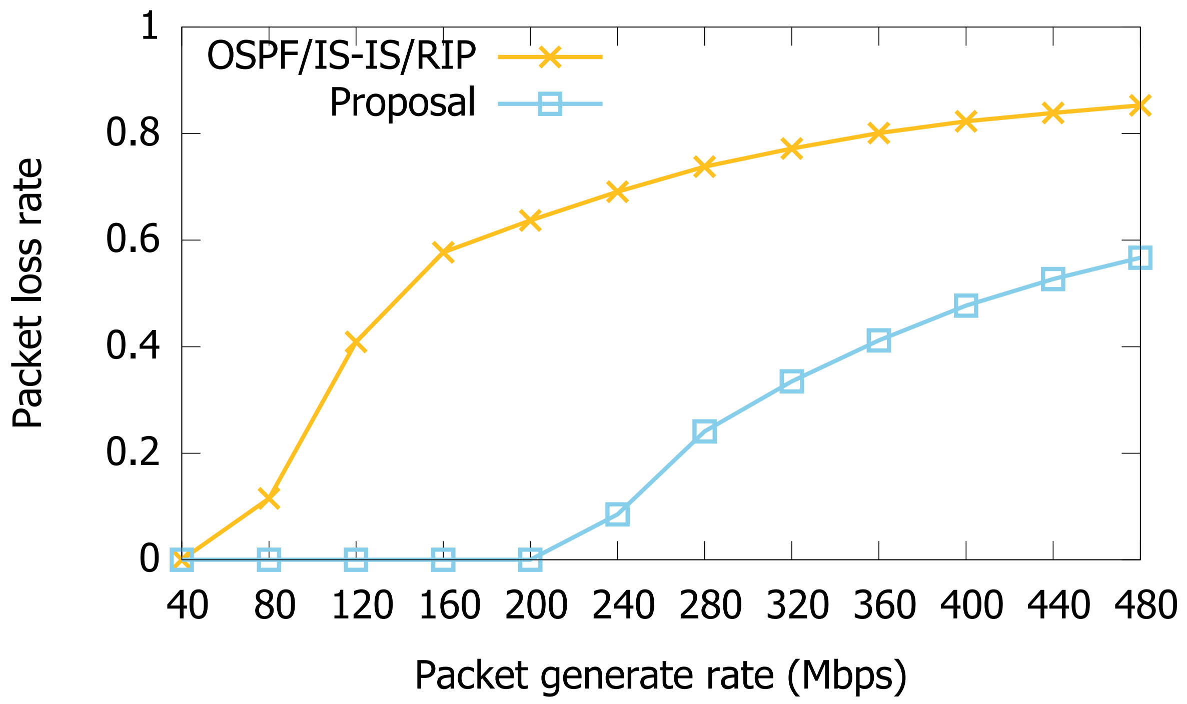
Furthermore, with such kind of bursty input traffic, we compare the performance of our proposal with existing routing protocols in different packet generation rates ranging from 40Mbps to 480Mbps. Fig. 6.6a demonstrates the comparison of packets loss rates between our proposed method and the conventional routing protocols after the simulation period of 1000s.

In addition to the comparison of the total throughput of the network, we compare the signaling overhead between our proposal with that of the conventional OSPF as shown in Fig. 6.6b. The result in the figure demonstrates that despite the communication overhead, our proposal outperforms the conventional routing protocol. In the conventional OSPF, each router needs to update its routing table by flooding a significantly large number of link-state advertisements (LSAs) during each updating period. On the other hand, in our proposal, instead of flooding LSAs to all routers, each router just sends its feature information to r_{sup} in each updating phase. After computing the routing strategy only, r_{sup} floods the strategy to other routers. This results in a significantly less signaling overhead than the conventional scenario whereby each router floods its LSAs to all routers.

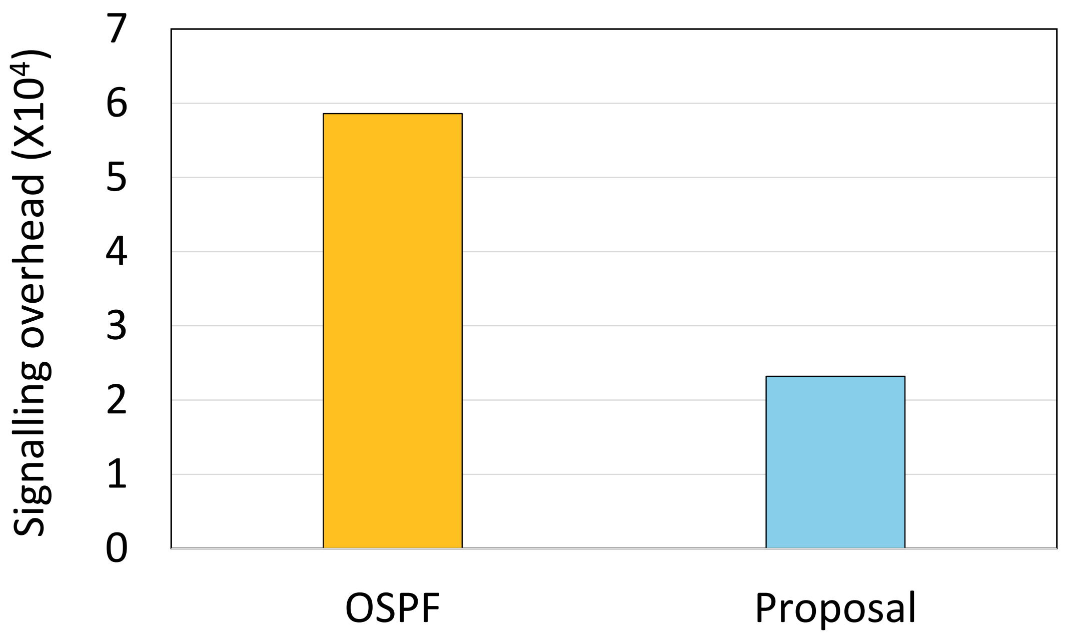
From the above results, it can be noticed that our proposal exhibits intelligent learning and therefore, achieves more effective network traffic control compared to the traditional routing methods. Furthermore, our proposal can be considered as a proof-of-concept demonstrating that the proposed technique is entirely independent of the existing routing protocols, and it can collect training data by itself over time to improve its performance. Indeed, such an approach can open up a new direction by providing an intelligent and self-routing decision making method in the future network traffic control systems.

6.4 Conclusion

Recently, deep learning emerged as a powerful machine learning technique for various disciplines such as computer science, mathematics, bio-engineering, and so forth. However, the deep learning application for network traffic allocation in wireless IoT networks is a relatively new research field. As the wireless networks especially the wireless IoT continue to become more complex, efficient network traffic control, particularly routing methodology, requires renewed research attention. In this vein, SDN has been viewed as the paradigm of next generation IoT due to its flexibility and conciseness. However, current SDN-IoT structure mainly utilizes conventional routing protocols based traffic allocations which are based on fixed rules and lacks the intelligence to learn from previous experiences. This can lead to the repetition of wrong traffic allocation decisions when similar traffic patterns happen. The inaccurate path decision results in the network congestion, which leads to further performance deterioration. In this Chapter, we propose a online deep learning based routing strategy which utilizes CNNs to choose the paths combinations according



(a) Average packet loss rate comparison for various packet generation rates.



(b) Signaling overhead comparison of our proposal with OSPF.

Figure 6.6: Comparison of our proposal and conventional OSPF, IS-IS, and RIP in terms of packet loss rate and signaling overhead.

to the network traffic trace in an online fashion. This strategy can not only better choose the paths combinations according to previous network trace, but also keeps improving its performance through continually learning from previous experience. Analysis shows that our proposal can avoid the congested paths and balance the network traffic, resulting in the significant improvement of packet loss rate and average packet delay in the SDN-IoT. Thus, it can be concluded that our proposal outperforms conventional routing protocols based traffic allocation in SDN-IoT.

Chapter 7

Conclusion

7.1 Summary and Discussions

The resource in IoT is limited. In order to better utilize the limited resources, we proposed a UAV-enabled IoT structure which employ UAV-enabled base station as the edge computing server in IoT. In the proposed IoT structure, the caching based D2D and UAV are jointly deployed to construct the heterogeneous IoT called as UAV-enabled IoT. With the considered UAV-enabled IoT, we considered the radio resource allocation in the network as an channel assignment problem since the number of orthogonal channels is limited and using overlapping channels in adjacent nodes with both primary and D2D links leads to severe interference. Furthermore, in the considered network, the mobility of IoT devices leads to highly dynamic situation which renders conventional channel assignment algorithms unsuitable. For overcoming those challenges, we presented an interference model and formulated a formal problem. Then, we proposed AC-POCA, a distributed Anti-Coordination game based algorithm for solving the channel assignment problem in the considered UAV-enabled IoT. Using AC-POCA, the IoT devices are able to use only local information to reach a steady state in the network. Through analysis, the uniqueness of the steady state in the proposed AC-POCA was also verified. Simulation results were provided to demonstrate that the proposal leads to fast convergence, low signaling overhead, and improved throughput in contrast with the existing channel allocation methods.

However, in the UAV-enabled IoT, the IoT devices are constructed heterogeneously and links between them are connected dynamically, which leads to unstable of the conventional IoT structure. Another big challenge of the heterogeneous IoT is that the tremendous number of object swilling to connect to the Internet should be considered in many underlying protocols. To address such problem, recently, the Software Defined Networking based IoT (SDN-IoT) is proposed to provide adequate solutions to heterogeneous underlying IoT devices. However, as the large scale devices in the SDN-IoT are heterogeneous which makes both the topology and traffic flows in the network high dynamically change. Current SDN structure mainly utilizes conventional channel assignment and routing protocols which are based on xed rules lacks the intelligence to catch on the high dynamically changed traffic patterns. To deal with the high dynamic changed network traffic, we proposed a novel deep learning based traffic load prediction method to predict the future traffic load and network congestion. Then, a deep learning based partially channel assignment algorithm (DLPOCA) was proposed to intelligently assign channels to each link in SDN-IoT. Finally, we combine the deep learning based traffic prediction and channel assignment to propose a novel intelligent channel assignment algorithm (TP-DLPOCA), which can intelligently avoid traffic congestion and quickly assign suitable channels to the wireless links of SDN-IoT. Extensive simulation results demonstrate that our proposal significantly outperforms the conventional channel assignment algorithms and ACPOCA.

Finally, with the allocated channel, we further proposed a deep learning based network traffic control algorithm to deal with the traffic allocation problem in the data plain of SDN-IoT. The a deep learning based network traffic control algorithm which utilizes CNNs to choose the paths combinations according to the network trafc trace in an online fashion. This strategy can not only better choose the paths combinations according to previous network trace, but also keeps improving its performance through continually learning from previous experience. Analysis shows that the proposal can avoid the congested paths and balance the network trafc, resulting in the signicant improvement of packet loss rate and average packet delay in the SDN-IoT. In summary, in the thesis, I research on the resource allocation problem in the IoT, the corresponding radio resource allocation and traffic allocation algorithm are proposed to alleviate the high burden and improve the network performance in IoT.

Bibliography

[1] Y. Kawamoto, N. Yamada, H. Nishiyama, N. Kato, Y. Shimizu, and Y. Zheng, "A feedback control-based crowd dynamics management in iot system," *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1466–1476, Oct 2017.

[2] Y. Kawamoto, H. Nishiyama, N. Kato, Y. Shimizu, A. Takahara, and T. Jiang, "Effectively collecting data for the location-based authentication in internet of things," *IEEE Systems Journal*, vol. 11, no. 3, pp. 1403–1411, Sept 2017.

[3] S. Verma, Y. Kawamoto, Z. M. Fadlullah, H. Nishiyama, and N. Kato, "A survey on network methodologies for real-time analytics of massive iot data and open research issues," *IEEE Communications Surveys Tutorials*, vol. 19, no. 3, pp. 1457–1477, thirdquarter 2017.

[4] J. Ni, K. Zhang, X. Lin, and X. Shen, "Securing fog computing for internet of things applications: Challenges and solutions," *IEEE Communications Surveys Tutorials*, vol. PP, no. 99, pp. 1–1, 2017.

[5] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of things: A survey on enabling technologies, protocols, and applications," *IEEE Communications Surveys Tutorials*, vol. 17, no. 4, pp. 2347–2376, Fourthquarter 2015.

[6] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, "Internet of things for smart cities," *IEEE Internet of Things Journal*, vol. 1, no. 1, pp. 22–32, Feb 2014.

[7] J. Gantz and D. Reinsel, "The digital universe in 2020: Big data, bigger digital shadows, and biggest growth in the far east," *IDC iView: IDC Analyze the future*, vol. 2007, no. 2012, pp. 1–16, 2012.

[8] S. Taylor, "The next generation of the internet revolutionizing the way we work, live, play, and learn," *CISCO, San Francisco, CA, USA, CISCO Point of View*, vol. 12, 2013.

[9] M. Z. Shafiq, L. Ji, A. X. Liu, J. Pang, and J. Wang, "A first look at cellular machine-to-machine traffic: large scale measurement and characterization," *ACM SIGMETRICS performance evaluation review*, vol. 40, no. 1, pp. 65–76, 2012.

[10] J. Manyika, M. Chui, J. Bughin, R. Dobbs, P. Bisson, and A. Marrs, *Disruptive technologies: Advances that will transform life, business, and the global economy.* McKinsey Global Institute San Francisco, CA, 2013, vol. 180.

[11] navigant, "https://www.navigantresearch.com/reports/iot-and-analytics-for-utilities-market-overview," *navigant research report.*

[12] M. Dohler, I. Vilajosana, X. Vilajosana, and J. Llosa, "Smart cities: An action plan," in *Proc. Barcelona Smart Cities Congress, Barcelona, Spain*, 2011, pp. 1–6.

[13] M. Ojo, D. Adami, and S. Giordano, "A sdn-iot architecture with nfv implementation," in *2016 IEEE Globecom Workshops (GC Wkshps)*, Dec 2016, pp. 1–6.

[14] I. Katzela and M. Naghshineh, "Channel assignment schemes for cellular mobile telecommunication systems: a comprehensive survey," *IEEE Personal Communications*, vol. 3, no. 3, pp. 10–31, June 1996.

[15] P. Tague, S. Nabar, J. A. Ritcey, and R. Poovendran, "Jamming-aware traffic allocation for multiple-path routing using portfolio selection," *IEEE/ACM Trans. Netw.*, vol. 19, no. 1, pp. 184–194, Feb. 2011. [Online]. Available: https://doi.org/10.1109/TNET.2010.2057515

[16] S. Kosta, A. Aucinas, P. Hui, R. Mortier, and X. Zhang, "Thinkair: Dynamic resource allocation and parallel execution in the cloud for mobile code offloading," in *2012 Proceedings IEEE INFOCOM*, March 2012, pp. 945–953.

[17] S. Mao, M. H. Cheung, and V. W. S. Wong, "An optimal energy allocation algorithm for energy harvesting wireless sensor networks," in *2012 IEEE International Conference on Communications (ICC)*, June 2012, pp. 265–270.

[18] P. Corcoran and S. K. Datta, "Mobile-edge computing and the internet of things for consumers: Extending cloud computing and services to the edge of the network," *IEEE Consumer Electronics Magazine*, vol. 5, no. 4, pp. 73–74, Oct 2016.

[19] I. Bucaille, S. Hthuin, A. Munari, R. Hermenier, T. Rasheed, and S. Allsopp, "Rapidly deployable network for tactical applications: Aerial base station with opportunistic links for unattended and temporary events absolute example," in *MILCOM(2013) IEEE Military Communications Conference*, Nov. 2013.

[20] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, "Unmanned aerial vehicle with underlaid device-to-device communications: Performance and tradeoffs," *IEEE Transactions on Wireless Communications*, vol. 15, no. 6, pp. 3949–3963, Jun. 2016.

[21] F. Tang, Z. M. Fadlullah, B. Mao, N. Kato, F. Ono, and R. Miura, "On a novel adaptive uav-mounted cloudlet-aided recommendation system for lbsns," *IEEE Transactions on Emerging Topics in Computing*, pp. 1–1, 2018.

[22] B. Bai, L. Wang, Z. Han, W. Chen, and T. Svensson, "Caching based socially-aware d2d communications in wireless content delivery networks: a hypergraph framework," *IEEE Wireless Communications*, vol. 23, no. 4, pp. 74–81, Aug. 2016.

[23] H. Nishiyama, M. Ito, and N. Kato, "Relay-by-smartphone: realizing multihop device-to-device communications," *IEEE Communications Magazine*, vol. 52, no. 4, pp. 56–65, April 2014.

[24] J. Liu, H. Nishiyama, N. Kato, and J. Guo, "On the outage probability of device-to-device-communication-enabled multichannel cellular networks: An rss-threshold-based perspective," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 1, pp. 163–175, Jan 2016.

[25] J. Liu, S. Zhang, N. Kato, H. Ujikawa, and K. Suzuki, "Device-to-device communications for enhancing quality of experience in software defined multi-tier lte-a networks," *IEEE Network*, vol. 29, no. 4, pp. 46–52, July 2015.

[26] J. Liu, Y. Kawamoto, H. Nishiyama, N. Kato, and N. Kadowaki, "Device-to-device communications achieve efficient load balancing in lte-advanced networks," *IEEE Wireless Communications*, vol. 21, no. 2, pp. 57–65, April 2014.

[27] N. Golrezaei, A. G. Dimakis, and A. F. Molisch, "Scaling behavior for device-to-device communications with distributed caching," *IEEE Transactions on Information Theory*, vol. 60, no. 7, pp. 4286–4298, Jul. 2014.

[28] K. Doppler, M. Rinne, C. Wijting, C. B. Ribeiro, and K. Hugl, "Device-to-device communication as an underlay to lte-advanced networks," *IEEE Communications Magazine*, vol. 47, no. 12, pp. 42–49, Dec. 2009.

[29] P. B. F. Duarte, Z. M. Fadlullah, A. V. Vasilakos, and N. Kato, "On the partially overlapped channel assignment on wireless mesh network backbone: A game theoretic approach," *IEEE Journal on Selected Areas in Communications*, vol. 30, no. 1, pp. 119–127, January 2012.

[30] Y. Liu, R. Venkatesan, and C. Li, "Channel assignment exploiting partially overlapping channels for wireless mesh networks," in *GLOBECOM 2009 - 2009 IEEE Global Telecommunications Conference*, Nov 2009, pp. 1–5.

[31] V. Bukkapatanam, A. A. Franklin, and C. S. R. Murthy, "Using partially overlapped channels for end-to-end flow allocation and channel assignment in wireless mesh networks," in *2009 IEEE International Conference on Communications*, June 2009, pp. 1–6.

[32] F. Kojima and S. Takahashi, "Anti-coordination games and dynamic stability," *International Game Theory Review*, vol. 9, no. 4, pp. 667–688, Dec. 2007.

[33] S. Al-Rubaye, E. Kadhum, Q. Ni, and A. Anpalagan, "Industrial internet of things driven by sdn platform for smart grid resiliency," *IEEE Internet of Things Journal*, vol. PP, no. 99, pp. 1–1, 2017.

[34] K. Sood, S. Yu, and Y. Xiang, "Software-defined wireless networking opportunities and challenges for internet-of-things: A review," *IEEE Internet of Things Journal*, vol. 3, no. 4, pp. 453–463, Aug 2016.

[35] Z. Qin, G. Denker, C. Giannelli, P. Bellavista, and N. Venkatasubramanian, "A software defined networking architecture for the internet-of-things," in *2014 IEEE Network Operations and Management Symposium (NOMS)*, May 2014, pp. 1–9.

[36] F. Lin, C. Chen, N. Zhang, X. Guan, and X. Shen, "Autonomous channel switching: Towards efficient spectrum sharing for industrial wireless sensor networks," *IEEE Internet of Things Journal*, vol. 3, no. 2, pp. 231–243, April 2016.

[37] R. Zhang, M. Wang, X. Shen, and L. L. Xie, "Probabilistic analysis on qos provisioning for internet of things in lte-a heterogeneous networks with partial spectrum usage," *IEEE Internet of Things Journal*, vol. 3, no. 3, pp. 354–365, June 2016.

[38] N. Zhang, N. Cheng, N. Lu, H. Zhou, J. W. Mark, and X. S. Shen, "Risk-aware cooperative spectrum access for multi-channel cognitive radio networks," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 3, pp. 516–527, March 2014.

[39] CISCO, "The Internet of Things Reference Model White Paper," *"http://cdn.iotwf.com/resources/71/IoT_Reference_Model_White_Paper_June_4_2014.pdf"*, (accessed March 2018).

[40] F. Al-Turjman, "Information-centric framework for the internet of things (iot): Traffic modeling and optimization," *Future Generation Computer Systems*, vol. 80, pp. 63 – 75, 2018.

[41] K. Akkaya and M. Younis, "A survey on routing protocols for wireless sensor networks," *Ad Hoc Networks*, vol. 3, no. 3, pp. 325 – 349, 2005.

[42] S. Tozlu, M. Senel, W. Mao, and A. Keshavarzian, "Wi-fi enabled sensors for internet of things: A practical approach," *IEEE Communications Magazine*, vol. 50, no. 6, pp. 134–143, June 2012.

[43] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "Openflow: Enabling innovation in campus networks," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69–74, Mar. 2008. [Online]. Available: http://doi.acm.org/10.1145/1355734.1355746

[44] I. F. Akyildiz, A. Lee, P. Wang, M. Luo, and W. Chou, "Research challenges for traffic engineering in software defined networks," *IEEE Network*, vol. 30, no. 3, pp. 52–58, May 2016.

[45] Z. M. Fadlullah, D. Takaishi, H. Nishiyama, N. Kato, and R. Miura, "A dynamic trajectory control algorithm for improving the communication throughput and delay in uav-aided networks," *IEEE Network*, vol. 30, no. 1, pp. 100–105, Jan. 2016.

[46] P. Zhan, K. Yu, and A. L. Swindlehurst, "Wireless relay communications with unmanned aerial vehicles: Performance and optimization," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 47, no. 3, pp. 2068–2085, Jul. 2011.

[47] S. Jeong, O. Simeone, and J. Kang, "Mobile edge computing via a uav-mounted cloudlet: Optimization of bit allocation and path planning," *IEEE Transactions on Vehicular Technology*, vol. PP, no. 99, pp. 1–1, 2017.

[48] C. Wang, F. R. Yu, C. Liang, Q. Chen, and L. Tang, "Joint computation offloading and interference management in wireless cellular networks with mobile edge computing," *IEEE Transactions on Vehicular Technology*, vol. PP, no. 99, pp. 1–1, 2017.

[49] K. Sood, S. Yu, Y. Xiang, and S. Peng, "Control layer resource management in sdn-iot networks using multi-objective constraint," in *2016 IEEE 11th Conference on Industrial Electronics and Applications (ICIEA)*, June 2016, pp. 71–76.

[50] B. Nguyen, N. Choi, M. Thottan, and J. V. der Merwe, "Simeca: Sdn-based iot mobile edge cloud architecture," in *2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, May 2017, pp. 503–509.

[51] S. A. Grandhi, R. D. Yates, and D. J. Goodman, "Resource allocation for cellular radio systems," *IEEE Transactions on Vehicular Technology*, vol. 46, no. 3, pp. 581–587, Aug 1997.

[52] C. Curescu and S. Nadjm-Tehrani, "Time-aware utility-based resource allocation in wireless networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 16, no. 7, pp. 624–636, July 2005.

[53] M. Dohler, A. Gkelias, and H. Aghvami, "Resource allocation for fdma-based regenerative multihop links," *IEEE Transactions on Wireless Communications*, vol. 3, no. 6, pp. 1989–1993, Nov 2004.

[54] H. Viswanathan and S. Mukherjee, "Performance of cellular networks with relays and centralized scheduling," *IEEE Transactions on Wireless Communications*, vol. 4, no. 5, pp. 2318–2328, Sep. 2005.

[55] Y. Liu, R. Hoshyar, X. Yang, and R. Tafazolli, "Integrated radio resource allocation for multihop cellular networks with fixed relay stations," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 11, pp. 2137–2146, Nov 2006.

[56] R. Yin, C. Zhong, G. Yu, Z. Zhang, K. K. Wong, and X. Chen, "Joint spectrum and power allocation for d2d communications underlaying cellular networks," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 4, pp. 2182–2195, April 2016.

[57] F. Wang, L. Song, Z. Han, Q. Zhao, and X. Wang, "Joint scheduling and resource allocation for device-to-device underlay communication," in *2013 IEEE Wireless Communications and Networking Conference (WCNC)*, April 2013, pp. 134–139.

[58] D. H. Lee, K. W. Choi, W. S. Jeon, and D. G. Jeong, "Two-stage semi-distributed resource management for device-to-device communication in cellular networks," *IEEE Transactions on Wireless Communications*, vol. 13, no. 4, pp. 1908–1920, April 2014.

[59] D. Feng, L. Lu, Y. Yuan-Wu, G. Y. Li, G. Feng, and S. Li, "Device-to-device communications underlaying cellular networks," *IEEE Transactions on Communications*, vol. 61, no. 8, pp. 3541–3551, August 2013.

[60] B. Zhou, H. Hu, S. Huang, and H. Chen, "Intracluster device-to-device relay algorithm with optimal resource utilization," *IEEE Transactions on Vehicular Technology*, vol. 62, no. 5, pp. 2315–2326, Jun 2013.

[61] M. Hasan, E. Hossain, and D. I. Kim, "Resource allocation under channel uncertainties for relay-aided device-to-device communication underlaying lte-a cellular networks," *IEEE Transactions on Wireless Communications*, vol. 13, no. 4, pp. 2322–2338, April 2014.

[62] X. Zhang, Z. Zheng, Q. Shen, J. Liu, X. S. Shen, and L. Xie, "Optimizing network sustainability and efficiency in green cellular networks," *IEEE Transactions on Wireless Communications*, vol. 13, no. 2, pp. 1129–1139, February 2014.

[63] Z. Zhou, M. Dong, K. Ota, J. Wu, and T. Sato, "Energy efficiency and spectral efficiency tradeoff in device-to-device (d2d) communications," *IEEE Wireless Communications Letters*, vol. 3, no. 5, pp. 485–488, Oct 2014.

[64] D. Wu, J. Wang, R. Q. Hu, Y. Cai, and L. Zhou, "Energy-efficient resource sharing for mobile device-to-device multimedia communications." *IEEE Trans. Vehicular Technology*, vol. 63, no. 5, pp. 2093–2103, 2014.

[65] Z. Zhou, M. Dong, K. Ota, G. Wang, and L. T. Yang, "Energy-efficient resource allocation for d2d communications underlaying cloud-ran-based lte-a networks," *IEEE Internet of Things Journal*, vol. 3, no. 3, pp. 428–438, June 2016.

[66] A. Raniwala, K. Gopalan, and T.-c. Chiueh, "Centralized channel assignment and routing algorithms for multi-channel wireless mesh networks," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 8, no. 2, pp. 50–65, 2004.

[67] T. R. Jensen and B. Toft, *Graph coloring problems.* John Wiley & Sons, 2011, vol. 39.

[68] D. S. Hochbaum, *Approximation algorithms for NP-hard problems.* PWS Publishing Co., 1996.

[69] P. Kyasanur and N. H. Vaidya, "Routing and interface assignment in multi-channel multi-interface wireless networks," in *IEEE Wireless Communications and Networking Conference, 2005*, vol. 4, March 2005, pp. 2051–2056 Vol. 4.

[70] M. K. Marina and S. R. Das, "A topology control approach for utilizing multiple channels in multi-radio wireless mesh networks," in *2nd International Conference on Broadband Networks, 2005.*, Oct 2005, pp. 381–390 Vol. 1.

[71] P. Bahl, R. Chandra, and J. Dunagan, "Ssch: Slotted seeded channel hopping for capacity improvement in ieee 802.11 ad-hoc wireless networks," in *Proceedings of the 10th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '04. New York, NY, USA: ACM, 2004, pp. 216–230. [Online]. Available: http://doi.acm.org/10.1145/1023720.1023742

[72] A. Raniwala and T. cker Chiueh, "Architecture and algorithms for an ieee 802.11-based multi-channel wireless mesh network," in *Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies.*, vol. 3, March 2005, pp. 2223–2234 vol. 3.

[73] H. Skalli, S. Ghosh, S. K. Das, L. Lenzini, and M. Conti, "Channel assignment strategies for multiradio wireless mesh networks: Issues and solutions," *IEEE Communications Magazine*, vol. 45, no. 11, pp. 86–95, November 2007.

[74] L. Zhou, X. Wang, W. Tu, G. Muntean, and B. Geller, “Distributed scheduling scheme for video streaming over multi-channel multi-radio multi-hop wireless networks,” *IEEE Journal on Selected Areas in Communications*, vol. 28, no. 3, pp. 409–419, April 2010.

[75] A. Mishra, V. Shrivastava, S. Banerjee, and W. Arbaugh, “Partially overlapped channels not considered harmful,” *SIGMETRICS Perform. Eval. Rev.*, vol. 34, no. 1, pp. 63–74, Jun. 2006. [Online]. Available: http://doi.acm.org/10.1145/1140103.1140286

[76] M. A. Hoque, X. Hong, and F. Afroz, “Multiple radio channel assignement utilizing partially overlapped channels,” in *GLOBECOM 2009 - 2009 IEEE Global Telecommunications Conference*, Nov 2009, pp. 1–7.

[77] P. B. F. Duarte, Z. M. Fadlullah, K. Hashimoto, and N. Kato, “Partially overlapped channel assignment on wireless mesh network backbone,” in *2010 IEEE Global Telecommunications Conference GLOBECOM 2010*, Dec 2010, pp. 1–5.

[78] Y. Ding, Y. Huang, G. Zeng, and L. Xiao, “Using partially overlapping channels to improve throughput in wireless mesh networks,” *IEEE Transactions on Mobile Computing*, vol. 11, no. 11, pp. 1720–1733, Nov 2012.

[79] P. Ciotrnae, F. G. Popescu, F. R. Enache, and M. G. Burgos, “Extended partially overlapped channel interference model and study for 802.11 wi-fi communications,” in *2014 International Conference on Applied and Theoretical Electricity (ICATE)*, Oct 2014, pp. 1–4.

[80] F. S. Bokhari and G. V. Zruba, “i-poca: Interference-aware partially overlapping channel assignment in 802.11-based meshes,” in *2013 IEEE 14th International Symposium on ”A World of Wireless, Mobile and Multimedia Networks” (WoWMoM)*, June 2013, pp. 1–6.

[81] T. Huynh, T. Onuma, K. Kuroda, M. Hasegawa, and W. Hwang, “Joint downlink and uplink interference management for device to device communication underlaying cellular networks,” *IEEE Access*, vol. 4, pp. 4420–4430, 2016.

[82] N. ul Hasan, W. Ejaz, I. Baig, M. Zghaibeh, and A. Anpalagan, “Qos-aware channel assignment for iot-enabled smart building in 5g systems,” in *2016 Eighth International Conference on Ubiquitous and Future Networks (ICUFN)*, July 2016, pp. 924–928.

[83] H. Kim, “Low power routing and channel allocation method of wireless video sensor networks for internet of things (iot),” in *2014 IEEE World Forum on Internet of Things (WF-IoT)*, March 2014, pp. 446–451.

[84] H. B. Salameh, S. Almajali, M. Ayyash, and H. Elgala, “Security-aware channel assignment in iot-based cognitive radio networks for time-critical applications,” in *2017 Fourth International Conference on Software Defined Systems (SDS)*, May 2017, pp. 43–47.

[85] D. Fudenberg and J. Tirole, "Game theory, 1991," *Cambridge, Massachusetts*, vol. 393, no. 12, p. 80, 1991.

[86] M. J. Osborne and A. Rubinstein, *A course in game theory.* MIT press, 1994.

[87] R. J. La and V. Anantharam, "Optimal routing control: Repeated game approach," *IEEE Transactions on Automatic Control*, vol. 47, no. 3, pp. 437–450, 2002.

[88] C. Busch, R. Kannan, and A. V. Vasilakos, "Approximating congestion+ dilation in networks via" quality of routing" games." *IEEE Trans. Computers*, vol. 61, no. 9, pp. 1270–1283, 2012.

[89] R. Kannan, C. Busch, and A. V. Vasilakos, "Optimal price of anarchy of polynomial and super-polynomial bottleneck congestion games," in *International Conference on Game Theory for Networks.* Springer, 2011, pp. 308–320.

[90] L. Gao, X. Wang, G. Sun, and Y. Xu, "A game approach for cell selection and resource allocation in heterogeneous wireless networks," in *Sensor, Mesh and Ad Hoc Communications and Networks (SECON), 2011 8th Annual IEEE Communications Society Conference on.* IEEE, 2011, pp. 530–538.

[91] S.-S. Byun, A. Vasilakos, and I. Balasingham, "An investigation of stochastic market equilibrium in cognitive radio networks," *IEEE Communications Letters*, vol. 14, no. 12, pp. 1122–1124, 2010.

[92] F. Meshkati, A. J. Goldsmith, H. V. Poor, and S. C. Schwartz, "A game-theoretic approach to energy-efficient modulation in cdma networks with delay qos constraints," *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 6, 2007.

[93] D. Niyato and E. Hossain, "Integration of ieee 802.11 wlans with ieee 802.16-based multihop infrastructure mesh/relay networks: A game-theoretic approach to radio resource management," *IEEE Network*, vol. 21, no. 3, 2007.

[94] L. Zhao, J. Zhang, and H. Zhang, "Using incompletely cooperative game theory in wireless mesh networks," *IEEE Network*, vol. 22, no. 1, 2008.

[95] Y. Song, C. Zhang, and Y. Fang, "Joint channel and power allocation in wireless mesh networks: A game theoretical perspective," *IEEE Journal on Selected Areas in Communications*, vol. 26, no. 7, pp. 1149–1159, Sep. 2008.

[96] W. Yuan, W. Liu, and W. Cheng, "Capacity maximization for variable-width wlans: A game-theoretic approach," in *2010 IEEE International Conference on Communications*, May 2010, pp. 1–5.

[97] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[98] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, p. 436, 2015.

[99] "The ibrain is here and it's already inside your phone." *https://backchannel.com/an-exclusive-look-at-how-ai-and-machine-learning-work-at-apple-8dbfb131932b#43bf9cm00*, accessed Jan. 2018.

[100] N. Kato, Z. M. Fadlullah, B. Mao, F. Tang, O. Akashi, T. Inoue, and K. Mizutani, "The deep learning vision for heterogeneous network traffic control: Proposal, challenges, and future perspective," *IEEE Wireless Communications*, vol. 24, no. 3, pp. 146–153, 2017.

[101] X. He, K. Wang, H. Huang, T. Miyazaki, Y. Wang, and S. Guo, "Green resource allocation based on deep reinforcement learning in content-centric iot," *IEEE Transactions on Emerging Topics in Computing*, pp. 1–1, 2018.

[102] Z. Wang, "The applications of deep learning on traffic identification," *BlackHat USA*, 2015.

[103] B. Mao, Z. M. Fadlullah, F. Tang, N. Kato, O. Akashi, T. Inoue, and K. Mizutani, "Routing or computing? the paradigm shift towards intelligent computer network packet transmission based on deep learning," *IEEE Transactions on Computers*, vol. 66, no. 11, pp. 1946–1960, Nov 2017.

[104] Z. Fadlullah, F. Tang, B. Mao, N. Kato, O. Akashi, T. Inoue, and K. Mizutani, "State-of-the-art deep learning: Evolving machine intelligence toward tomorrow's intelligent network traffic control systems," *IEEE Communications Surveys Tutorials*, vol. PP, no. 99, pp. 1–1, 2017.

[105] M. A. Hoque, X. Hong, and F. Afroz, "Multiple radio channel assignement utilizing partially overlapped channels," in *GLOBECOM 2009 - 2009 IEEE Global Telecommunications Conference*, Nov 2009, pp. 1–7.

[106] Z. Feng and Y. Yang, "Characterizing the impact of partially overlapped channel on the performance of wireless networks," in *IEEE GLOBECOM 2008 - 2008 IEEE Global Telecommunications Conference*, Nov 2008, pp. 1–6.

[107] ——, "How much improvement can we get from partially overlapped channels?" in *2008 IEEE Wireless Communications and Networking Conference*, March 2008, pp. 2957–2962.

[108] Monderer and Shapley, "Potential games," *Journal of Games and Economic Behavior*, vol. 14, pp. 124–143, May 1996.

[109] V. Srivastava, J. Neel, A. Mackenzie, R. Menon, L. Dasilva, J. Hicks, J. Reed, and R. Gilles, "Using game theory to analyze wireless ad hoc networks," *IEEE Commun. Surveys Tutorials*, vol. 7, no. 4, pp. 46 – 56, Jan. 2005.

[110] A. Matsui and K. Matsuyama, "An approach to equilibrium selection," *Journal of Economic Theory*, vol. 65, no. 2, pp. 415 – 434, Apr. 1995.

[111] K. Michihiro and R. Rob, "Bandwagon effects and long run technology choice," *Games and Economic Behavior*, vol. 22, no. 1, pp. 30 – 60, Jan. 1998.

[112] F. Kojima and S. Takahashi, "Anti-coordination games and dynamic stability," *International Game Theory Review*, vol. 9, no. 4, pp. 667–688, Dec. 2007.

[113] E. Koutsoupias and C. H. Papadimitriou, "Worst-case equilibria," *Computer Science Review*, vol. 3, no. 2, pp. 65–69, Apr. 2009.

[114] W. Zhao, H. Nishiyama, Z. Fadlullah, N. Kato, and K. Hamaguchi, "Dapa: Capacity optimization in wireless networks through a combined design of density of access points and partially overlapped channel allocation," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 5, pp. 3715–3722, May 2016.

[115] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds. Curran Associates, Inc., 2015, pp. 91–99. [Online]. Available: http://papers.nips.cc/paper/5638-faster-r-cnn-towards-real-time-object-detection-with-region-proposal-networks.pdf

[116] Y. Kim, "Convolutional neural networks for sentence classification," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, 2014, p. 17461751.

[117] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3d convolutional networks," in *2015 IEEE International Conference on Computer Vision (ICCV)*, Dec 2015, pp. 4489–4497.

[118] F. Tang, B. Mao, Z. M. Fadlullah, N. Kato, O. Akashi, T. Inoue, and K. Mizutani, "On removing routing protocol from future wireless networks: A real-time deep learning approach for intelligent traffic control," *IEEE Wireless Communications*, vol. 25, no. 1, pp. 154–160, February 2018.

[119] "WILL API," *https://scarsty.gitbooks.io/will/content/*, (accessed December 2017).

[120] F. Tang, Z. M. Fadlullah, N. Kato, F. One, and R. Miura, "Ac-poca: Anti-coordination game based partially overlapping channels assignment in combined uav and d2d based networks," *IEEE Transactions on Vehicular Technology*, vol. PP, no. 99, pp. 1–1, 2017.

[121] J. A. C. Fuertes, M. Philipp, and E. Baccelli, "Routing across wired and wireless mesh networks: Experimental compound internetworking with ospf," in *2012 8th International Wireless Communications and Mobile Computing Conference (IWCMC)*, Aug 2012, pp. 739–745.

[122] A. Banerjee, J. Drake, J. P. Lang, B. Turner, K. Kompella, and Y. Rekhter, "Generalized multiprotocol label switching: an overview of routing and management enhancements," *IEEE Communications Magazine*, vol. 39, no. 1, pp. 144–150, Jan 2001.

[123] B. Fortz and M. Thorup, "Internet traffic engineering by optimizing ospf weights," in *Proceedings IEEE INFOCOM 2000. Conference on Computer Communications. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (Cat. No.00CH37064)*, vol. 2, March 2000, pp. 519–528 vol.2.

[124] ——, "Optimizing ospf/is-is weights in a changing world," *IEEE Journal on Selected Areas in Communications*, vol. 20, no. 4, pp. 756–767, May 2002.

[125] T. G. Griffin, F. B. Shepherd, and G. Wilfong, "The stable paths problem and interdomain routing," *IEEE/ACM Transactions on Networking*, vol. 10, no. 2, pp. 232–243, April 2002.

Publications

Journals

[1] Fengxiao Tang, Zubair Md. Fadlullah, Nei Kato, Fumie Ono, and Ryu Miura, "AC-POCA: Anti-Coordination Game based Partially Overlapping Channels Assignment in Combined UAV and D2D based Networks," IEEE Transactions on Vehicular Technology, vol. 67, no. 2, pp. 1672-1683, Feb. 2018.

[2] Fengxiao Tang, Bomin Mao, Zubair Md. Fadlullah, and Nei Kato, "On a Novel Deep Learning Based Intelligent Partially Overlapping Channel Assignment in SDN-IoT," IEEE Communications Magazine, vol. 56, no. 9, pp. 80-86, Sep. 2018.

[3] Fengxiao Tang, Zubair Md. Fadlullah, Bomin Mao, Nei Kato, Fumie Ono, and Ryu Miura, "On A Novel Adaptive UAV-Mounted Cloudlet-Aided Recommendation System for LBSNs," IEEE Transactions on Emerging Topics in Computing, In press, DOI: 10.1109/TETC.2018.2792051.

[4] Fengxiao Tang, Bomin Mao, Zubair Md. Fadlullah, Nei Kato, Osamu Akashi, Takeru Inoue, and Kimihiro Mizutani, "On Removing Routing Protocol from Future Wireless Networks: A Real-time Deep Learning Approach for Intelligent Traffic Control, " IEEE Wirelesss Magazine (WCM), vol. 25, no. 1, pp. 154-160, Feb. 2018.

[5] Fengxiao Tang, Zubair Md. Fadlullah, Bomin Mao, and Nei Kato, "An Intelligent Traffic Load Prediction Based Adaptive Channel Assignment Algorithm in SDN-IoT: A Deep Learning Approach," IEEE Internet of Things Journal, In press. DOI : 10.1109/JIOT.2018.2838574.

[6] Zubair Md. Fadlullah, Fengxiao Tang, Bomin Mao, Jiajia Liu and Nei Kato, "On Intelligent Traffic Control for Large-Scale Heterogeneous Networks: A Value Matrix-Based Deep Learning Approach," in IEEE Communications Letters, vol. 22, no. 12, pp. 2479-2482, Dec. 2018.

[7] Zubair Md. Fadlullah, Fengxiao Tang, Bomin Mao, Nei Kato, Osamu Akashi, Takeru Inoue, and Kimihiro Mizutani, "State-of-the-Art Deep Learning: Evolving Machine Intelligence Toward Tomorrows Intelligent Network Traffic Control Systems," IEEE Communications Surveys and Tutorials, vol. 19, no. 4, pp. 2432-2455, May 2017.

[8] Bomin Mao, Fengxiao Tang, Zubair Md. Fadlullah, Nei Kato, Osamu Akashi, Takeru Inoue, and Kimihiro Mizutani, "A Novel Non-supervised Deep Learning

Based Network Traffic Control Method for Software Defined Wireless Networks," IEEE Wireless Communications Magazine, vol. 25, no. 4, pp. 74-81, Aug. 2018.

[9] Zubair Md. Fadlullah, Bomin Mao, Fengxiao Tang, and Nei Kato, "Value Iteration Architecture based Deep Learning for Intelligent Routing Exploiting Heterogeneous Computing Platforms," IEEE Transactions on Computers, In press, DOI : 10.1109/TC.2018.287448.

[10] Bomin Mao, Zubair Md. Fadlullah, Fengxiao Tang, Nei Kato, Osamu Akashi, Takeru Inoue, and Kimihiro Mizutani, "Routing or Computing? The Paradigm Shift Towards Intelligent Computer Network Packet Transmission Based on Deep Learning, " IEEE Transactions on Computers, vol. 66, no. 11, pp. 1946-1960, Nov. 2017.

[11] Nei Kato, Zubair Md. Fadlullah, Bomin Mao, Fengxiao Tang, Osamu Akashi, Takeru Inoue, and Kimihiro Mizutani, "The Deep Learning Vision for Heterogeneous Network Traffic Control Proposal, Challenges, and Future Perspective ," IEEE Wireless Communications, vol. 24, no. 3, pp. 146-153, Dec. 2016.

[12] Nei Kato, Zubair Md. Fadlullah, Fengxiao Tang, Bomin Mao, Shigenori Tani, Atsushi Okamura, and Jiajia Liu, "Optimizing Space-Air-Ground Integrated Networks by Artificial Intelligence," IEEE Wireless Communications Magazine (WCM), Accepted.

Refereed Conference Papers

[13] Fengxiao Tang, Bomin Mao, Zubair Md. Fadlullah, and Nei Kato, "Deep Spatiotemporal Partially Overlapping Channel Allocation: Joint CNN and Activity Vector Approach," IEEE Global Communications Conference (GLOBECOM 2018), Abu Dhabi, UAE, Accepted.

[14] Fengxiao Tang, Bomin Mao, Zubair Md. Fadlullah, and Nei Kato, "On Extracting the Spatial-Temporal Features of Network Traffic Patterns: A Tensor Based Deep Learning Model," to appear, IEEE International Conference on Network Infrastructure and Digital Content (IC-NIDC18), to be held in Huaxi District, Guiyang, China, Aug. 2018.

[15] Bomin Mao, Zubair Md. Fadlullah, Fengxiao Tang, Nei Kato, Osamu Akashi, Takeru Inoue, and Kimihiro Mizutani, "A Tensor Based Deep Learning Technique for Intelligent Packet Routing," IEEE Global Communications Conference (GLOBECOM 2017), Singapore, Dec. 2017.

Grants

1) Best Paper Award, IEEE GLOBECOM18 Conference (Dec. 2018)

2) Best Paper Award, IEEE IC-NIDC17 Conference (Aug. 2018)

3) Best Paper Award, IEEE GLOBECOM17 Conference (Dec. 2017)