AN INTELLIGENT TRAFFIC LOAD PREDICTION BASED ADAPTIVE CHANNEL ASSIGNMENT ALGORITHM IN SDN-IOT

Seminar Report

Submitted in partial fulfillment of the requirements for the award of degree of

BACHELOR OF TECHNOLOGY

In

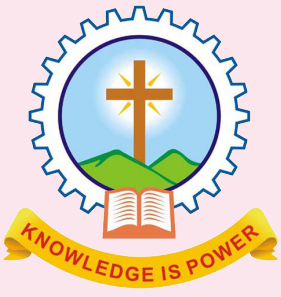
COMPUTER SCIENCE AND ENGINEERING

of

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

Submitted By

A T SIDHARTH



Department of Computer Science & Engineering
Mar Athanasius College Of Engineering
Kothamangalam

AN INTELLIGENT TRAFFIC LOAD PREDICTION BASED ADAPTIVE CHANNEL ASSIGNMENT ALGORITHM IN SDN-IOT

Seminar Report

Submitted in partial fulfillment of the requirements for the award of degree of

BACHELOR OF TECHNOLOGY

In

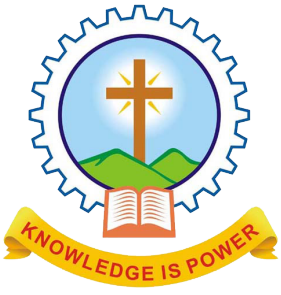
COMPUTER SCIENCE AND ENGINEERING

of

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

Submitted By

A T SIDHARTH



Department of Computer Science & Engineering
Mar Athanasius College Of Engineering
Kothamangalam

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
MAR ATHANASIUS COLLEGE OF ENGINEERING
KOTHAMANGALAM

CERTIFICATE

This is to certify that the report entitled **An Intelligent Traffic Load Prediction Based Adaptive Channel Assignment Algorithm in SDN-IoT** *submitted by* **Mr. A T SIDHARTH**, *Reg.No.***MAC15CS001** *towards partial fulfillment of the requirement for the award of Degree of Bachelor of Technology in Computer science and Engineering from APJ Abdul Kalam Technological University for December 2018 is a bonafide record of the seminar carried out by him under our supervision and guidance.*

.....	.....	.....
Prof. Joby George	**Prof. Neethu Subash**	**Dr. Surekha Mariam Varghese**
Faculty Guide	***Faculty Guide***	***Head Of Department***

Date:

Dept. Seal

ACKNOWLEDGEMENT

First and foremost, I sincerely thank the 'God Almighty' for his grace for the successful and timely completion of the seminar.

I express my sincere gratitude and thanks to Dr. Solly George, Principal and Dr. Surekha Mariam Varghese, Head Of the Department for providing the necessary facilities and their encouragement and support.

I owe special thanks to the staff-in-charge Prof. Joby george, Prof. Neethu Subash and Prof. Joby Anu Mathew for their corrections, suggestions and sincere efforts to co-ordinate the seminar under a tight schedule.

I express my sincere thanks to staff members in the Department of Computer Science and Engineering who have taken sincere efforts in helping me to conduct this seminar.

Finally, I would like to acknowledge the heartfelt efforts, comments, criticisms, co-operation and tremendous support given to me by my dear friends during the preparation of the seminar and also during the presentation without whose support this work would have been all the more difficult to accomplish.

ABSTRACT

Due to the fast increase of sensing data and quick response requirement in the Internet of Things (IoT) delivery network the high speed transmission has emerged as an important issue. Due to this growth the high dynamics of traffic load make the conventional fixed channel assignment algorithm ineffective. As a solution to this issue a deep learning based traffic load prediction algorithm can be used to forecast future traffic load and congestion in network. The predicted traffic load value can be used to assign channels to each link in an IoT through a Deep Learning based Partially Channel Assignment Algorithm referred to as TP-DLPOCA. Software Defined Network-IoT architecture and partially overlapping channels are considered to improve the transmission quality. The two neural network structures considered in the algorithm are CNN (Convolutional Neural Network) and DBA (Deep Belief Architecture). The evaluation results of this intelligent channel assignment significantly outperforms the conventional channel assignment algorithms.

Contents

List of Figures

List of Tables

List of Abbreviations

IOT	Internet Of Things
SDN	Software Defined Network
POC	Partially Overlapping Channel
TL	Traffic Load
IF	Interference Factor
CNN	Convolutional Neural Network
DBA	Deep Belief Architrcture
DLPOCA	Deep Learning based Partially Overlapping Channel Assignment
ACPOCA	Anti Coordination Partially Overlapping Channel Assignment
CTP	Centralized Traffic Prediction
S-CTP	Semi Centralized Traffic Prediction
DTP	Distributed Traffic Prediction
CoCAG	Cooperative Channel Assignment Game

Introduction

Real world Internet of Things (IoT) deployments are fundamentally heterogeneous. Software Defined Networking (SDN)[1] is a famous technique used in the IoT to deal with heterogeneous resources and structures. In such SDN-IoT, heterogeneous devices sense and collect data in the sensing plane, and then send the data to the gateway after integration through switches in the data plane. With the increasing number of devices, the load of integrated traffic in switches may become significantly heavy, and multiple channels are needed to be evenly assigned to each link to balance the load. Since high interference exists between non-orthogonal channels and the number of orthogonal channels is limited, the Partially Overlapping Channel (POC)[2] can be a good solution to decrease interference and improve network throughput[3]. In the dynamic traffic load scenario, the traffic load of each load may change frequently. With this high dynamics of the current 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.

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[4]. 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). A kind of temperature sensing device may collect 3kB temperature once in every 30s, while 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. Due to this growth the high dynamics of traffic load make the conventional fixed channel assignment algorithm ineffective.

Existing system

There exists several conventional channel assignment algorithms. 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.The 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. Most of them focused on the traffic changes in the long-term and did not consider the traffic load changes caused by routing. The switches or routers in the IoT system is made to exchange their channel states. These states are maintained in matrix in order to assign channel to each link. Due to the increase in number of devices, this signalling creates overhead traffic load.

Anti coordination partially overlapping channel assignment(ACPOCA)[5] is one such conventional method, which uses partially overlapped channel in the system. POC[2] contributes to spectrum utilization and improves the bandwidth available to the network users. The use of overlapping channels leads to better performance in contrast to three non-overlapping channels for wireless networks. Thus in this system POC are used and the output data of ACPOCA[5] is used to train the neural network for channel assignment in this proposed system. Games in which players score the highest when they choose opposite strategies are called anti-coordination games. The objective of the game is to maximize the network throughput. However, in the anti coordination game, a player only focuses on his utility. This utility can be a proportional measure of the connectivity of each node. But the main drawback this system is that the channel is assigned based on the current traffic load presented for each link.

Proposed system

Initially a powerful deep learning based algorithm is used to predict the complex traffic, which can achieve above 90 percent accuracy and have a quick response time.Then with the centralized SDN control as shown in figure 3.1, the deep learning based traffic prediction and channel assignment are combined, that uses the predicted traffic load as the criterion to perform the intelligent channel assignment.

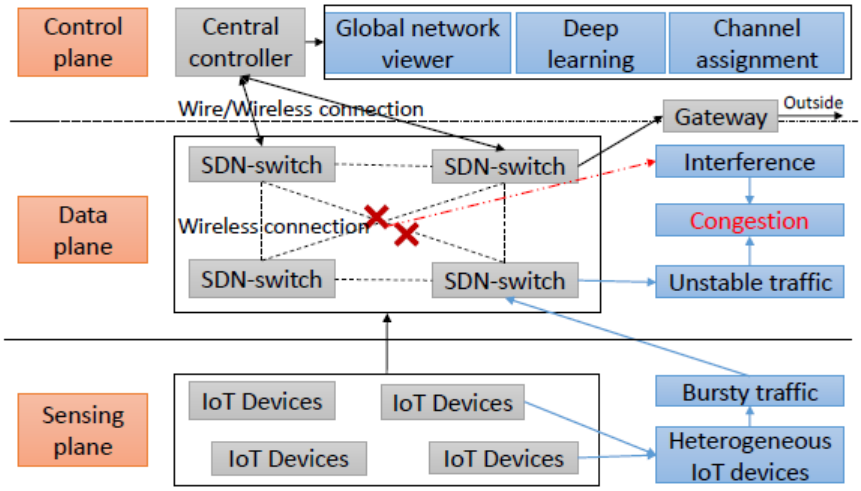


Figure 3.1: The SDN-Iot architecture

3.1 System model

The considered system model can be approached through three models: network model, interference model and training mode.

3.1.1 Network model

Consider the SDN-IoT[1] 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 G = (D U S U C,E) to represent the network where D denotes the set of devices in the network and D = $\{d_1, d_2, ...d_{|D|}\}$. And S denotes the set of switches and S =$\{s_1, s_2, ..., s_M\}$ 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 area served by the AP. The average number of devices belong to each switch area is presented as R, namely $|D|$ = M.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 3.1. 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 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.

3.1.2 Interference model

In a multi-channel deployment network, the interference between each channel is critical to the network capacity. In this section, we present an interference range vector based model to state those interferences. Consider the channel interference between channels is affected by both geographical distance and distance in the spectrum[6]. In order to calculate the interference affected by geographical distance, an interference range vector, shown in Table I, is used to measure the interference level of channels . Here δ refers to the interference range for a channel separation between channels p and q, IR(δ) = $|p - q|$ denotes the geographical interference distance between channels p and q.

Table 3.1: Interference Range (IR)

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

Then, consider the different geographical distance between assigned channels. We use a weight metric Interference Factor $f_{p,q}$ to denote the interference level between two active channels p and q assigned to links. 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:

1. $f_{p,q}$ = 0: when $\delta \geq 5$ or d $>$ IR(δ). When the nodes are assigned orthogonal channels or have enough distance to avoid interference, no interference occurs between the radios.

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

   $f_{p,q} = IR(\delta)/d$ (Eqn: 1)

3. $f_{p,q} = \infty$: when 0$<$ δ $<$5 and d = 0. This happens because of the self interference problem. Hence, two overlapping channels ($\delta < 5$) are not viable to be assigned to the node due to their full interference.

Here, $f_{p,q}$ is used to measure the interference level of channels, and is the main parameter used in the POC assignment algorithm to judge the quality of POCs. 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 fp;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.

3.1.3 Training 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).

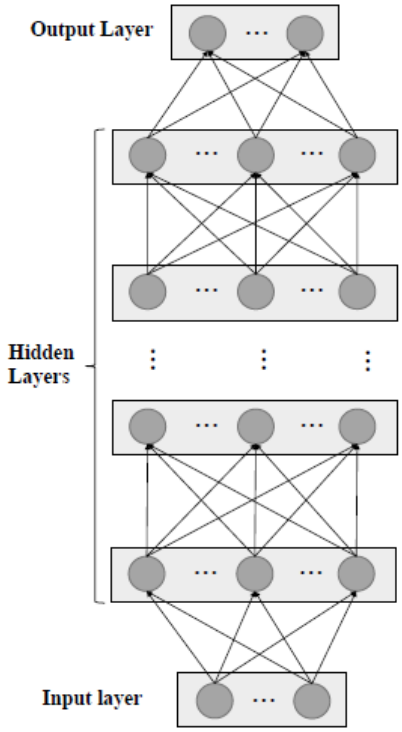


Figure 3.2: Considered DBA structure

As shown in Fig. 3.2, 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. $w_{i,j}$ 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. When trained

on a set of examples without supervision, a DBN can learn to probabilistically reconstruct its inputs. The layers then act as feature detectors. After this learning step, a DBN can be further trained with supervision to perform classification. 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. The forward propagation process can be modeled as a log-likelihood function,

$$l(w, b, x_{input}, y_{output}) = \sum_{t=1}^{m} logp(v^t) \qquad (Eqn : 2)$$

where, 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(v(t)) to represent the probability of 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 back propagation process, the gradient descent method is adopted to adjust the link weight w and bias b, which is represented as:

$$w = w + \eta \frac{\partial(w, b, x_{input}, y_{output})}{\partial w} \qquad (Eqn : 3.1)$$

$$b = b + \eta \frac{\partial(w, b, x_{input}, y_{output})}{\partial b} \qquad (Eqn : 3.2)$$

where,η is the learning rate of training process. The second considered deep learning structure is the deep Convolutional Neural Network(CNN) as shown in Fig 3.3. 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.

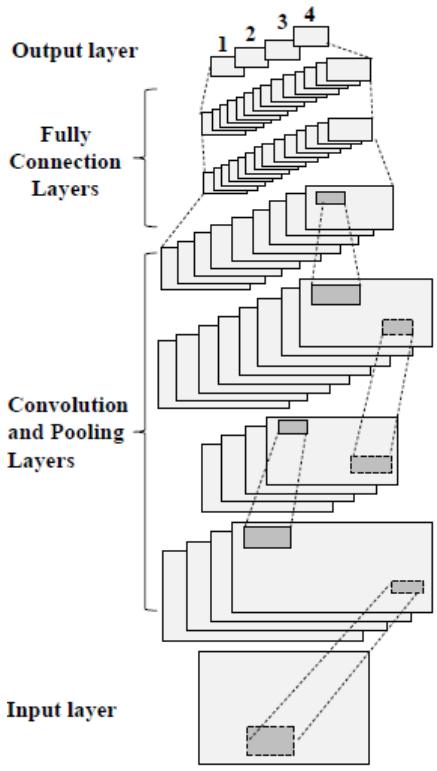


Figure 3.3: Considered Deep CNN structure

A CNN consists of an input and an output layer, as well as multiple hidden layers. The hidden layers of a CNN typically consist of convolutional layers, pooling layers, fully connected layers and normalization layers. The covolutional layers are good at capture the spatial and temporal connections of the input data[4]. Convolutional layers apply a convolution operation to the input, passing the result to the next layer. The convolution emulates the response of an individual neuron to visual stimuli. Each convolutional neuron processes data only for its receptive field.

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. Convolutional networks may include local or global pooling layers, which combine the outputs of neuron clusters at one layer into a single neuron in the next layer.

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 $W^{(l1)}$ to denote the filters and the k^{th} filter is represented by $W_k^{(l1)}$, the obtained feature map by the convolution operation can be shown as follows.

$$u_{i,j,k}^{(l1)} = (U^{(l1)} * W_k^{(l1)})(i,j) + w_{bk}^{(l1)} \qquad (Eqn: 4.1)$$

$$= \sum_{p=1}^{P} \sum_{m=1}^{M'} \sum_{n=1}^{N'} w_{m,n,p} a_{i+m,j+n}^{(l-1)} + w_{bk}^{(l1)} \qquad (Eqn: 4.2)$$

$$a_{i,j,k}^{(l1)} = f(u_{i,j,k}^{(l1)}) \qquad (Eqn: 4.3)$$

where f() is the activation function and $a_{i,j,k}^{(l1)}$ is the activated value of the unit in the i_th row and j_th column of the feature map. Therefore, $u_{i,j,k}^{(l1)}$ is the value before activation. $w_{bk}^{(l1)}$ denotes the bias of the k_th filter and is usually a single numeric value.$a_{i+m,j+n}^{(l-1)}$ 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. Fully connected layers connect every neuron in one layer to every neuron in another layer. It is in principle the same as the traditional multi-layer perceptron neural network. Then, the similar forward propagation and back propagation processes are repeated to fine-tune the whole CNN structure.

3.2 Deep learning based traffic load prediction

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 simi-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 ealier, 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_{int} + TL_{rel}.

3.2.1 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.

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_k^i. Then, the past traffic loads TL_i of switch i are formed as a length-N vector, $TL^i = \{tl_k^i, tl_{k-1}^i ..., i_{k-N+1}\}$ 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.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_k^i. Then, the past traffic loads TL^i of switch i are formed as a length-N vector, $TL^i = \{tl_k^i, tl_{k-1}^i ..., tl_{k-N+1}^i\}$,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 from 1 to M, and formats them as a traffic load matrix TL $= \{TL^1, TL^2, ..., TL^M\}$. 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 = $\{tl_k, tl_{t-1} ..., tl_{k-N+1}\}$. 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 $tl_{k+1} = \{tl_{k+1}^1, tl_{k+1}^2, ..., tl_{k+1}^M\}$ 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.

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[7]. 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.

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. The output of all deep-CNNs is recorded as $TLP_{k+1} = \{tlp^1_{k+1}, tlp^2_{k+1}, ..., tlp^M_{k+1}\}$. The real future traffic load of time slot (k + 1) is recorded as $TL_{k+1} = \{tl^1_{k+1}, tl^2_{k+1} ..., tl^M_{k+1}\}$. Therefore, we can calculate the prediction accuracy according to the following equation.

$$\frac{1}{KxM} \sum_{k=0}^{K-1} \sum_{i=1}^{M} \frac{|tlp^i_{k+1} - tl^i_{k+1}|}{tl^i_{max}} \qquad (Eqn : 5)$$

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.

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.

3.2.2 Traffic load prediction in semi-central control system:SCTP

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.

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 t_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 .

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 a switch i, the training data of its local neural network can be

represented as$(x_{input}, y_{output}) = (TL_int^i, tl_int^i_{k+1})$. 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 DBN to perform this training process. As mentioned in the data collection phase, the trained DBA will be utilized to predict the future integrated traffic load which is represented as $tlp\_int^i_{k+1}$, 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. The input of training data can be formed as $(tl_k, tlp\_int_{k+1}) = (\{tl^1_k, tl^2_k, ..., tl^M_k\}), \{tlp\_int^1_{k+1}, ..., tlp\_int^M_{k+1}\}$ 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
$(x_{input}, y_{output}) = ((tl_k, tlp\_int_{k+1}), tl^i_{k+1})$

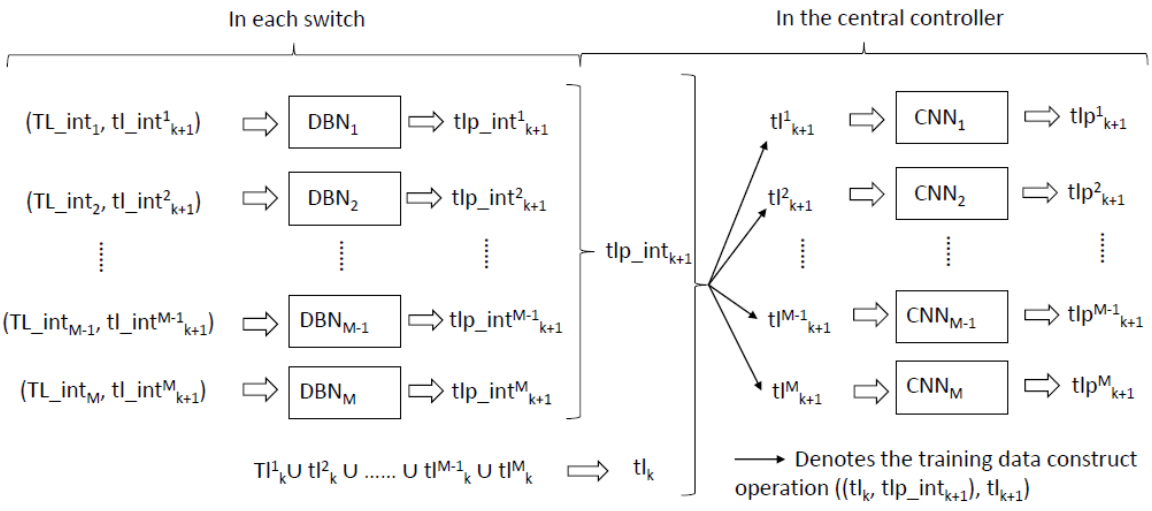


Figure 3.4: The training and prediction phase in a semi-central control system.

.

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.3.4.

3.2.3 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^i_{k+1}, is taken as the output. Thus, the training data can be represented as $(x_{input}, y_{output}) = ((TL\_rel^i, TL\_int^i), tl^i_{k+1})$.

The second kind of input is only the combined traffic load TL^i. In this case, the training data can be denoted as $(x_{input}, y_{output}) = (TL^i, tl^i_{k+1})$. And in the simulation, 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 85in 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 at the end.

3.3 Proposed deep learning based partially overlapping 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 problems mentioned earlier the conventional channel assignment algorithms cannot meet the new requirement of the future SDN-IoT. Aided by the high computation ability in the future SDN, 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 proposal, a deep learning method is used to train the network with the data from conventional anti-coordination game based partially channel assignment algorithm (AC-POCA), in which, the partially overlapping channels are assigned to each link by using an anti-coordination game. In the 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.

Table 3.2: Learning performance with different combination of features

	HC	IF	HC+IF	TL	HC+IF+TL
Accuracy	23%	24%	67%	100%	100%
Epochs	∞	∞	∞	500	800

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, we can find that any feature combination containing the feature of traffic load can get 100 percentage accuracy. To the opposite extreme, the accuracy of using any combination without traffic load is less than 70 percentage. 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.

The channel assignment algorithm into two parts. In the first part, a Deep Learning based Partially Overlapping Channel Assignment algorithm (DLPOCA) is considered. In the second part, 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 is considered.

3.3.1 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.

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

3.3.2 Deep learning based channel assignment jointed with prediction

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 TPDLPOCA. 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.Since, the traffic load of all switches in next slot is predicted and formatted as $tlp_k{+}1 = \{tlp^1_{k+1}, tlp^2_{k+1}, ..., tlp^M_{k+1}\}$ 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.

Deep learning based channel assignment

The whole assignment process can be divided into two steps, i.e., the training phase and dynamic channel assignment phase.

• Training phase:

The traffic load and channel assignment result of AC-POCA are used as the training data set. Before using the data set, training data is characterized into a suitable format.The traffic load is used as the main feature to construct the input of training data. Such training data format is denoted as $tl_k = \{tl_k^1, tl_k^2, ..., tl_k^M\}$. 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.

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, the complex output will significantly decrease the training accuracy. Therefore, as the same method employed in our traffic prediction algorithm, M x 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 = $\{l_1, l_2, ..., l_{C_max}|l \in \{0, 1\}\}$. And if channel i is assigned, the value of the ith 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, different kinds of neural network structures and different parameters for training are tried. The comparison of the training results of different structures and parameters. 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 $\{WM_i, j|i \leq M, j \leq E_{max}\}$.

• 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 channelL_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.

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 the proposed deep learning based channel assignment, only one iteration is needed, which is the main reason the proposal can significantly outperform the conventional one.

Deep learning based channel assignment jointed with prediction

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 TPDLPOCA. 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.Since, the traffic load of all switches in next slot is predicted and formated as $tlp_{k+1} = \{tlp^1_{k+1}, tlp^2_{k+1}, .., tlp^M_{k+1}\}$, 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.

3.4 Performance evaluation

Evaluation is done from three aspects: the prediction accuracy, the performance of DLPOCA, and the performance of TP-DLPOCA.

3.4.1 Prediction accuracy

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. There are different kind of devices At first, evaluation of accuracy in performance of traffic load prediction with three proposed mechanisms, namely, CTP, SCTP, and DTP is measured. In the case of PIC, the number of switches and the slot length are 16 and 1s, respectively.

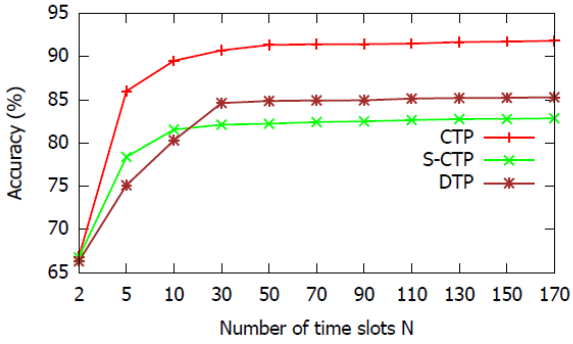


Figure 3.5: The prediction accuracy with different numbers of time slots, N

Comparison is done for the prediction accuracy of three mechanisms with different number of slots N, which is an important parameter of the prediction algorithm as shown in fig 4.1.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%).

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.

3.4.2 Performance of deep learning based channel assignment

The learning performance of POC with different learning structures and different learning parameters are considered. Then the comparison of the POC accuracy of our proposal. Finally, comparison of the throughput between othe proposed DLPOCA and traditional channel assignment algorithms (i.e., the orthogonal channel assignment, POC, AC-POCA). After running all those training processes with mini-batch size of 20 and 500 epoches, the accuracy result is shown in Fig 4.2.

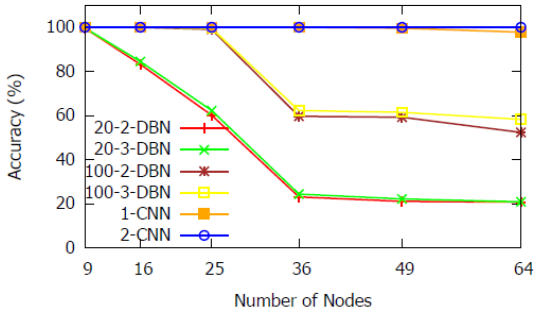


Figure 3.6: The accuracy with different configuration of learning structure.

From the result it made clear 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 the network and is chosen as the training structure.Thus the proposed DLPOCA chooses the deep-CNN as the training structure.

3.4.3 Performance of the joint deep learning based prediction and channel assignment

To further improve the channel performance, the intelligent deep learning based channel assignment with deep learning based traffic load prediction was combined. The result of the below Fig. 4.3 (a) and (b) shows the performance of the proposed TP-DLPOCA with conventional channel assignment algorithms (i.e., CoCAG, AC-POCA).

(a)

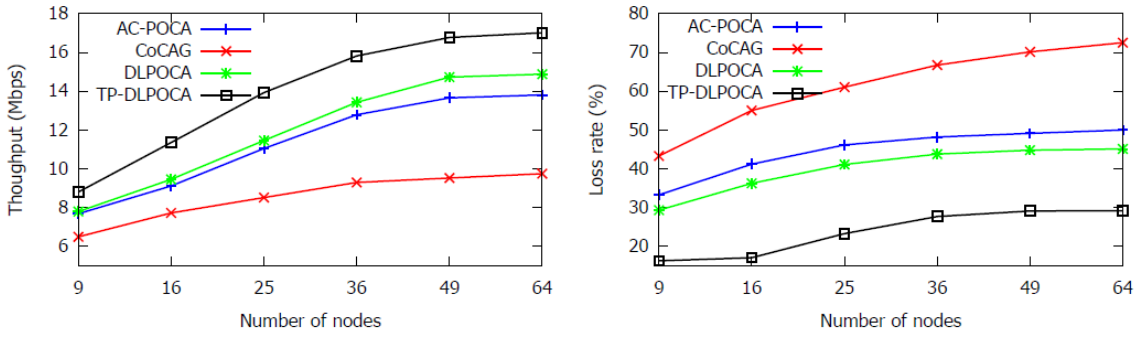


Figure 3.7: (a).The convergence compared with proposal and conventional algorithm (b).The throughput compared with proposed DLPOCA and conventional algorithms.

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. This good performance of the proposed TP-DLPOCA can be credited as the online intelligent channel assignment strategy.

Conclusion

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. But the proposed method of combination of the deep learning based traffic prediction and intelligent channel assignment algorithm (TP-DLPOCA) can intelligently avoid traffic congestion and quickly assign suitable channels to the wireless links of SDN-IoT. Extensive simulation results demonstrate that the proposal significantly outperforms the conventional channel assignment algorithms.

References

[1] 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.*

[2] 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 Internet of Things Journal, vol. 4, no. 5, pp. 1466– 1476, Oct 2017.*

[3] 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.*

[4] 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.*

[5] F. Tang, Z. M. Fadlullah, N. Kato, F. One, and R. Miura,,"Ac-poca: Anticoordination 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.*

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

[7] 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.*