

An Intelligent Traffic Load Prediction-Based Adaptive Channel Assignment Algorithm in SDN-IoT: A Deep Learning Approach

Fengxiao Tang^{1b}, *Student Member, IEEE*, Zubair Md. Fadlullah^{1b}, *Senior Member, IEEE*,
Bomin Mao^{1b}, *Student Member, IEEE*, and Nei Kato^{1b}, *Fellow, IEEE*

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. Assigning suitable channels in the wireless IoT delivery network is a basic guarantee of high speed transmission. However, the high dynamics of traffic load (TL) make the conventional fixed channel assignment algorithm ineffective. Recently, the software defined networking-based IoT (SDN-IoT) is proposed to improve the transmission quality. Besides this, the intelligent technique of deep learning is widely researched in high computational SDN. Hence, we first propose a novel deep learning-based TL prediction algorithm to forecast future TL and congestion in network. Then, a deep learning-based partially channel assignment algorithm is proposed to intelligently allocate channels to each link in the SDN-IoT network. Finally, we consider a deep learning-based prediction and partially overlapping channel assignment to propose a novel intelligent channel assignment algorithm, 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.

Index Terms—Deep learning, Internet of Things (IoT), partially overlapping channel assignment (POCA), software defined network (SDN), traffic load (TL) prediction.

I. INTRODUCTION

REAL-WORLD Internet of Things (IoT) [1]–[4] deployments are fundamentally heterogeneous. Software defined networking (SDN) [5], [6] is a famous technique used in the IoT to deal with heterogeneous resources and structures [7]. In such SDN-based IoT (SDN-IoT) as depicted in Fig. 1, 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 [8]–[10]. Since high interference

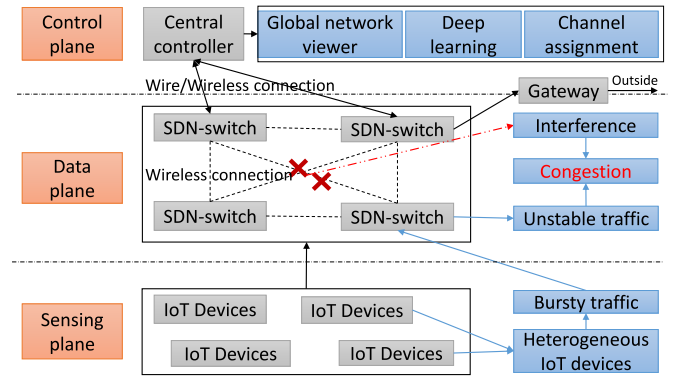


Fig. 1. SDN-IoT architecture.

exists between nonorthogonal 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 [11]–[13]. However, current POC algorithms (POCAs) 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 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. To solve this problem, in our previous work [14], an anti-coordination-based POCA (AC-POCA) was proposed, which can efficiently reduce the iteration times of channel assignment process, and improve the network throughput. However, without a central controller, both signaling and suspension time of the network are limited by the distributed setting. Therefore, to address such challenges, in the first part of this paper, a deep learning-based, intelligent POCA algorithm with the centralized SDN is proposed. The *contributions* of the deep learning-based proposal, in the first part of this paper, can be explained in two aspects.

- 1) 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.

Manuscript received February 28, 2018; revised April 27, 2018; accepted May 12, 2018. Date of publication May 21, 2018; date of current version January 16, 2019. This work was supported by the Japan Society for the Promotion of Science (JSPS) KAKENHI under Grant 16H05858. (Corresponding author: Nei Kato.)

The authors are with the Graduate School of Information Sciences, Tohoku University, Sendai 980-8579, Japan (e-mail: fengxiao.tang@it.is.tohoku.ac.jp; zubair@it.is.tohoku.ac.jp; bomin.mao@it.is.tohoku.ac.jp; kato@it.is.tohoku.ac.jp).

Digital Object Identifier 10.1109/JIOT.2018.2838574

- 2) 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., AC-POCA), the channel assignment can be finished in just single iteration.

In summary, this approach, which we refer to as the deep learning-based channel assignment (DLCA), can efficiently reduce the suspension time caused by channel assignment, and achieves almost nonsuspending flows during the channel assignment process.

Additionally, in existing channel assignment algorithms, as the most important baseline metric in the channel assignment, the traffic load (TL) is usually assumed to be continuous and stable. This means that the TL in the next time interval after the channel assignment is similar to that in last time interval. However, the real TLs 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: 1) periodic sensing; 2) event-driven sensing; and 3) query-based sensing [15]–[18]. 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 3-kB temperature data once in every 30 s, another kind of humidity sensing device may collect 10-kB humidity data once every 1 min. Those different policies result in highly complex, periodically bursty distribution of the TL. For the event-driven and query-based sensing devices, the TL 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 TL prediction, most of them focused on the traffic changes in the long-term and did not consider the TL changes caused by routing. Therefore, in the second part of this paper, a deep learning-based prediction and POCA algorithm is proposed. The *contributions* of the second part of this paper are separately outlined next.

- 1) First, we use the powerful deep learning approach to predict the complex traffic, which can achieve above 90% accuracy and have a quick response time (5 ms<).
- 2) Second, we investigate the advantage of using the centralized SDN technique in the deep learning-based TL prediction in the IoT environment. In order to show the improvement of deep learning-based TL

prediction in SDN-IoT compare with conventional IoT, we, respectively, design three TL prediction algorithms to suit three different control systems (i.e., centralized SDN control system, semicentralized 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.

- 3) Finally, with the centralized SDN control, we combine the deep learning-based traffic prediction and channel assignment, that uses the predicted TL as the criterion to perform the intelligent channel assignment. Such proposed intelligent channel assignment, which we refer to as TP-DLCA, can efficiently increase the channel assignment accuracy and processing speed of channel assignment. The simulation results demonstrate that both throughput and delay in the SDN-IoT with our proposal are better than those of the conventional algorithms.

The remainder of this paper is organized as follows. Section II surveys related works. Section III presents the considered network architecture, interference model, and training model. Then, in Section IV, we propose a deep learning-based TL prediction method by considering different network control systems. Then, we propose the DLCA algorithm, and combine it with deep learning-based TL prediction in Section V-A. The performance evaluation of our proposals are presented in Section VI. Finally, Section VII concludes this paper.

II. RELATED RESEARCH WORKS

The SDN-IoT structure was first proposed by Qin *et al.* [7] which incorporates and supports commands in a heterogeneous structure to optimize the SDN-IoT network. After the first SDN-IoT structure was proposed, many research works emerged. Sood *et al.* [19] employed a multiobjective constraint to manage the layer resource in SDN-IoT. Ojo *et al.* [20] presented an SDN-IoT architecture with network function virtualization implementation to address the new challenges of IoT. Nguyen *et al.* [21] proposed an SDN-based IoT mobile edge cloud architecture to deploy diverse IoT services at the mobile edge.

The work in [22] proposed a new channel assignment strategy based on nonoverlapping channels, and demonstrated how this contributes to spectrum utilization and improves the bandwidth available to the network users. On the other hand, the works in [12] and [13] demonstrated that the use of overlapping channels leads to better performance in contrast to three nonoverlapping channels for wireless networks. Following this finding, new heuristic channel assignment algorithms were proposed in [23]. In our earlier work in [11], instead of heuristics, an optimal channel assignment exploiting POCs for wireless mesh networks was proposed. After that, we further proposed an AC-POCA [14], in which, the nodes use only local information to play the game, and can reach a steady state quickly. Moreover, since this AC-POCA is dynamic, it is suitable to be employed in the fixed or dynamic topology network.

Our earlier work in [24] 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. The survey conducted in [25] demonstrated that there exist different deep learning architectures such as deep Boltzmann machines, deep convolutional neural networks (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 [26] 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-accelerated SDR enabling massively parallel computing for the deep learning was shown to substantially improve the backbone network traffic control. However, similar to the aforementioned researches, this paper also adopted a supervised deep learning system dependent on a conventional rule-based routing method. Then, our previous work in [27] proposed a real-time deep learning approach for intelligent traffic control in the wireless network exploiting deep-CNNs with uniquely characterized inputs and outputs to intelligently control the network packets forwarding. However, all of the previous researches focused on the intelligent packets forwarding process and did not take into account the intelligent channel assignment problem in the wireless network.

III. SYSTEM MODEL

In this section, we describe our considered network model, interference and training process.

A. 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 $D = \{d_1, d_2, \dots, d_{|D|}\}$ and S denotes the set of switches and $S = \{s_1, s_2, \dots, s_M\}$, 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 multihop 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 in Fig. 1.

TABLE I
IR

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

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 10-kB data in every 30 s and another kind of periodic sensing device collects 7-kB data in every 20 s.

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.

B. Interference Model

In a multichannel deployment network, the interference between each channel is critical to the network capacity. In this section, we present an interference range (IR) vector-based model to state those interferences.

Consider the channel interference between channels is affected by both geographical distance and distance in the spectrum [11]. In order to calculate the interference affected by geographical distance, an IR vector, shown in Table I, is used to measure the interference level of channels [28]. Here δ refers to the IR for a channel separation between channels p and q , $IR(\delta) = |p - q|$ denotes the geographical interference distance between channels p and q .

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 $d > IR(\delta)$. 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 IR. In this case, IF should be a ratio proportional to the distance between the nodes. IF can be calculated as follows:

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

- 3) $f_{p,q} = \infty$: when $0 \leq \delta < 5$ and $d = 0$. This happens because of the self interference problem. Hence,

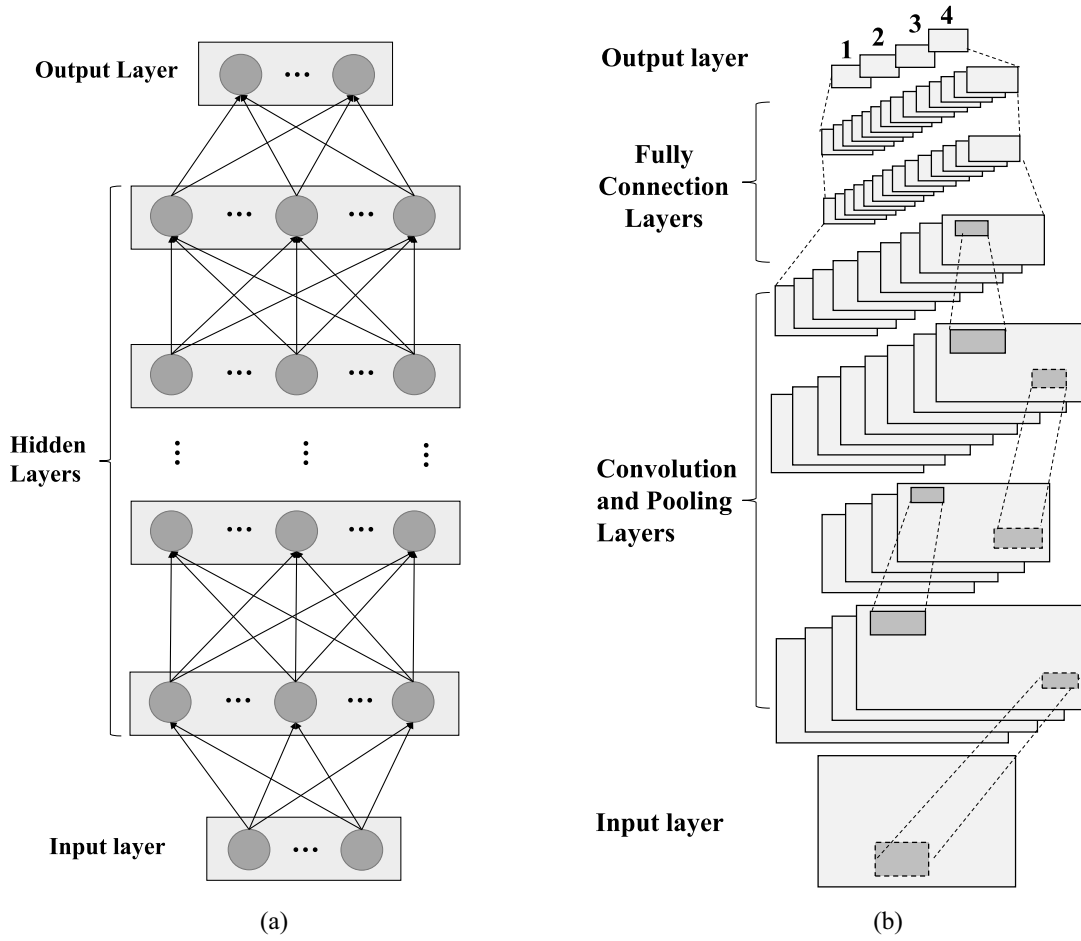


Fig. 2. Employed deep learning structures. Considered (a) DBA structure and (b) deep CNN structure.

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 POCA 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 $f_{p,q}$ 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 DLCA algorithm, the IMatrix is no longer needed, each switch just receives TL information and activates neural network weight matrix obtained by the training process. The only signaling overhead is the TL transmission process between switch and central controller. Next, we describe the deep learning training model used in training process.

C. Training Model

In our proposed training process, we consider two neural network structures, the basic deep belief architecture (DBA) and the deep-CNN. As shown in Fig. 2(a), 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 in [26], the forward propagation process can be modeled as a log-likelihood function

$$l(w, b, x_{\text{input}}, y_{\text{output}}) = \sum_{t=1}^m \log p(\mathbf{v}^{(t)}) \quad (2)$$

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

Since the purpose of the training process is to maximize $l(w, b, x_{\text{input}}, y_{\text{output}})$, in the backpropagation process, the

gradient descent method is adopted to adjust the link weight w and bias b , which is represented as

$$w = w + \eta \frac{\partial l(w, b, x_{\text{input}}, y_{\text{output}})}{\partial w} \quad (3)$$

$$b = b + \eta \frac{\partial l(w, b, x_{\text{input}}, y_{\text{output}})}{\partial b} \quad (4)$$

where η is the learning rate of training process.

The second considered deep learning structure is the deep-CNN as shown in Fig. 2(b). 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 [29], [30]. In the deep-CNN, the convolutional layers are good at capture the spatial and temporal connections of the input data [31]. 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:

$$\begin{aligned} u_{i,j,k}^{(l_1)} &= \left(U^{(l_1-1)} * W_k^{(l_1)} \right)(i, j) + w_{bk}^{(l_1)} \\ &= \sum_{p=1}^P \sum_{m=1}^{M'} \sum_{n=1}^{N'} w_{m,n,p} a_{i+m,j+n,p}^{(l_1-1)} + w_{bk}^{(l_1)} \end{aligned} \quad (5)$$

$$a_{i,j,k}^{(l_1)} = f(u_{i,j,k}^{(l_1)}) \quad (6)$$

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. $a_{i+m,j+n,p}^{(l_1-1)}$ 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.

IV. PROPOSED 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 central controller, which is a totally centralized control system. In order to research the performance between using centralized SDN system and semicentralized or distributed conventional control system without centralized SDN, we separately design our deep learning-based TL 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 semicentral control system. Based on these three different control systems, we propose three deep learning-based TL prediction methods, namely, central control-based traffic load prediction (CTP), semicentral CTP (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 TL is the arrival traffic flow. As mentioned earlier, in each switch, the TL sequence, TL, consists of two parts: 1) the relayed traffic flow from other switches denoted by TL_rel and 2) the integrated traffic flow composed by the sensing data from devices in the sensing plane denoted by TL_int. Therefore, $TL = TL_{\text{rel}} + TL_{\text{int}}$.

A. Traffic Load Prediction in Central Control System

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.

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 TL sequence, TL, of every switch in the last N time slots. And the length of each time slot is represented as Δ . The TL of switch i in last time slot k is recorded as tl_k^i . Then, the past TLs TL ^{i} of switch i are formed as a length- N vector, $TL^i = \{tl_k^i, tl_{k-1}^i, \dots, tl_{k-N+1}^i\}$, 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 TL series of every switch, and formats them as a TL matrix $TL = \{TL^1, TL^2, \dots, TL^M\}$. From the point of the time series, the TLs of all switches in the last N time slots can be also represented as $TL = \{tl_k, tl_{k-1}, \dots, tl_{k-N+1}\}$.

After data collection, the TL matrix TL is used as the input of training data. In the next time slot, the

central controller records the TL as the real future TL $tl_{k+1} = \{tl_{k+1}^1, tl_{k+1}^2, \dots, tl_{k+1}^M\}$ 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.

2) *Training Phase*: In this phase, in order to obtain a better training performance, we use a deep-CNN to fit our matrix-based training data [27]. In our earlier work [24], 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 TL 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 TL of one switch. Thus, the central controller only uses the future TL of one switch as the output of corresponding deep-CNN. For example, the training data of deep-CNN CNN^i is $(x_{input}, y_{output}) = (TL, tl_{k+1}^i)$. Then, the central controller trains all the deep-CNNs, respectively, to obtain all the stable weight matrices.

3) *Prediction and Accuracy Calculation Phase*: In the prediction phase, the central controller undertakes the future TL 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 TL, which is a forward propagation process as mentioned in Section III-C. The output of all deep-CNNs is recorded as $TLP_{k+1} = \{tlp_{k+1}^1, tlp_{k+1}^2, \dots, tlp_{k+1}^M\}$. As mentioned above, the real future TL of time slot $(k+1)$ is recorded as $TL_{k+1} = \{tl_{k+1}^1, tl_{k+1}^2, \dots, tl_{k+1}^M\}$. Therefore, we can calculate the prediction accuracy according to the following equation:

$$\frac{1}{K \times M} \sum_{k=0}^{K-1} \sum_{i=1}^M \frac{|tlp_{k+1}^i - tl_{k+1}^i|}{tl_{max}^i} \quad (7)$$

where K represents the total number of considered time slots. tl_{max}^i represents the maximum TLs of switch i . Here, we simply consider the maximum TL is equal to the maximum buffer size of the switch.

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 TL data, and the training phase is processed periodically with the collected new training data. Therefore, the weight matrices are periodically adjusted.

B. Traffic Load Prediction in Semicentral Control System

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 preprediction 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 preprediction information from all switches.

1) *Data Collection Phase*: In the central control system, the central controller predicts the TL 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 semicentral 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 semicentral control system, switches should perform some pretreatment 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 TL tl_k^i of the last time slot, and also separately records the relayed TL TL_rel^i and integrated TL TL_int^i of the last N time slots. Then, each switch i predicts the future integrated TL $tlp_int_{k+1}^i$ 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 TL 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 .

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 TL with its past N -time-slot integrated TLs. Therefore, for switch i , the training data of its local neural network can be represented as $(x_{input}, y_{output}) = (TL_int^i, tlp_int_{k+1}^i)$. 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) mentioned in Section III-C to perform this training process. As we mentioned in the data collection phase, the trained DBA will be utilized to predict the future integrated TL 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 TL tl_k and predicted integrated TL tlp_int_{k+1} of all switches. Since the TL and integrated traffic are two different network features, we form them as two channels of the input data, similar to our earlier research in [27]. Therefore, the input of training data can be formed as a matrix $(tl_k, tlp_int_{k+1}) = (\{tl_k^1, tl_k^2, \dots, tl_k^M\}, \{tlp_int_{k+1}^1, tlp_int_{k+1}^2, \dots, tlp_int_{k+1}^M\})$. As mentioned earlier, the deep learning structures in the central controller are utilized to predict the future TLs of all switches. Similar to the central control-based prediction, we utilize M deep-CNNs to make the

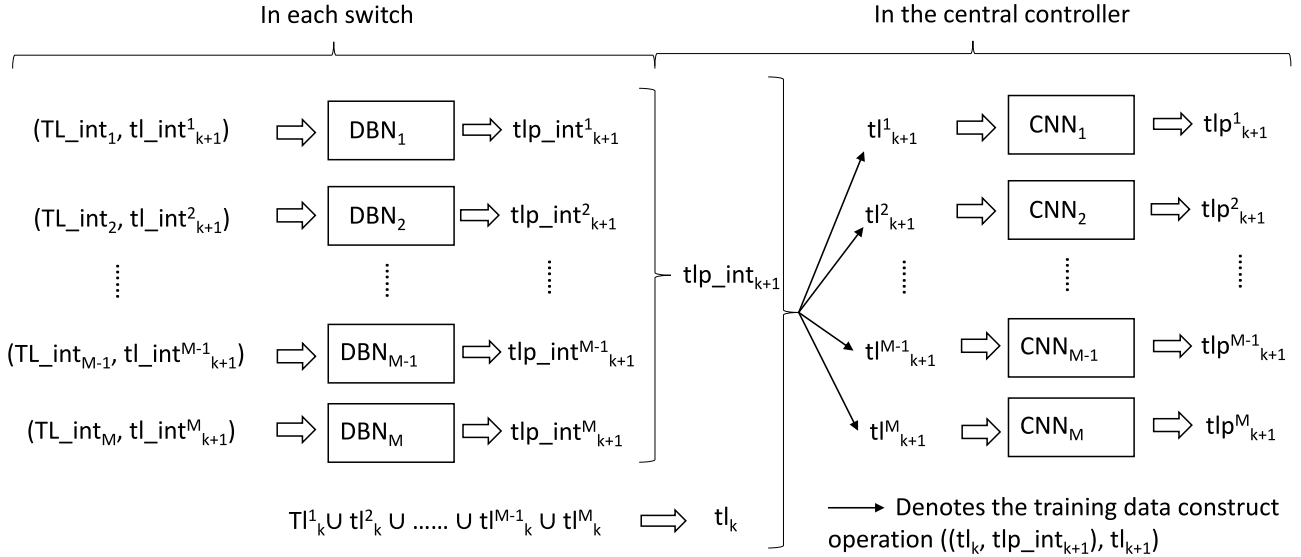


Fig. 3. Training and prediction phase in a semicentral control system.

prediction to alleviate the computational burden and guarantee the accuracy. Therefore, for CNN^i , its labeled training data is formed as $(x_{input}, y_{output}) = ((tl_k, tlp_int_{k+1}), tlp_{k+1}^i)$.

Except the above-mentioned two phases, the prediction phase and online phase in the semicentral 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.

C. Traffic Load Prediction in Distributed Control System

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 TL prediction method is designed as follows.

In the distributed control system, each switch only collects its own TL including the relayed TL and integrated TL. Without additional information of other switches, the relationship between two kinds of TLs 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 semicentral 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 TL and relayed TL as input. Therefore, switch i records the integrated TL TL_int^i and relayed TL TL_rel^i of the last N time slots. Then, the two TLs are constructed to a two channel matrix as input of training data. Correspondingly, the TL 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 TL 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 Section VI), 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 this paper.

The CTP, S-CTP, 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 VI.

V. PROPOSED DEEP LEARNING-BASED PARTIALLY CHANNEL ASSIGNMENT

After the TLs 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 Section I, 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 [26], we propose a new DLCA algorithm, which shows better convergence performance than the conventional algorithms, and leads to better network throughput.

In our proposal, we use deep learning to train the network with the data from our prior research [14], in which, the POCs 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 DLCA 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 AC-POCA with fixed topology, TL is the

TABLE II
LEARNING PERFORMANCE WITH DIFFERENT
COMBINATION OF FEATURES

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

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 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 II, we can find that any feature combination containing the feature of TL can get 100% accuracy. To the opposite extreme, the accuracy of using any combination without TL is less than 70%. Besides the accuracy rate of different combinations, we compare the training epochs (i.e., training time) of the combinations containing TL. And the result shows that the method only using TL as input of training data can get the best performance of training. Therefore, in the training process, the TL 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 POCA algorithm (DLPOCA). In the second part, we further propose an intelligent DLCA strategy which joints the DLPOCA with traffic load prediction algorithm (TP-DLPOCA) to obtain further improved performance.

A. 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.

1) *Training Phase*: Here, we use the TL 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 TL as the main feature to construct the input of training data. Such training data format is denoted as $tl_k = \{tl_k^1, tl_k^2, \dots, tl_k^M\}$. Because the AC-POCA only considers the current TL, and the result is not affected by the TL of the past time sequence, we only use the TL of the last one slot as the input of training data.

Then, we consider that the assigned channel 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 [24], 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 $\{NN_{ij} | i \leq M, j \leq E_{\max}\}$. For each neural network, the output is characterized as a vector consisting of C_{\max} binary elements, which can be denoted as $L = \{l_1, l_2, \dots, l_{C_{\max}-1}, l_{C_{\max}} | l \in \{0, 1\}\}$. And if channel i is assigned, the value of the i th element is 1, otherwise 0. Therefore, the training data of each neural network is indicated as $(x_{\text{input}}, y_{\text{output}}) = (tl_k, L)$.

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 Section VI-B. 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 $\{WM_{ij} | i \leq M, j \leq E_{\max}\}$.

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 TL information tl_k to each switch periodically, and each switch uses the current TL 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 Algorithm 1.

With DLCA, the channel assignment result can be simply obtained via a forward propagation process, which is much faster than 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 DLCA, only one iteration is needed, which is the main reason why our proposal can significantly outperform the conventional one.

B. 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) TL. This assumption is reasonable when the TL is constantly and slowly changed. However, in the practical environment,

Algorithm 1 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**

Algorithm 2 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_{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 1.

the TL is not so smooth. And in different applications, the TL may suddenly change or have more complex features. To solve this problem, we further combine the DLCA with the proposed TL prediction method, which is named as TP-DLPOCA. To compare it with DLPOCA, we replace the input TL in the training data with the predicted TL obtained via the deep learning structures described in Section IV. Since, the TL of all switches in next slot is predicted and formatted

TABLE III
CONFIGURATION OF PIC AND EIC

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

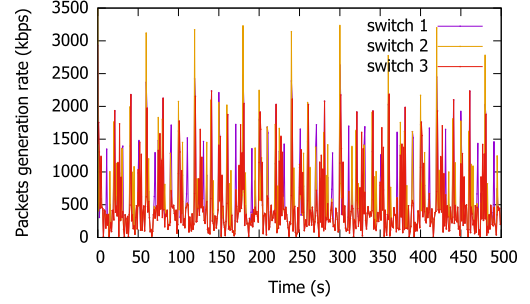


Fig. 4. Integrated traffic pattern of different switches in the PIC.

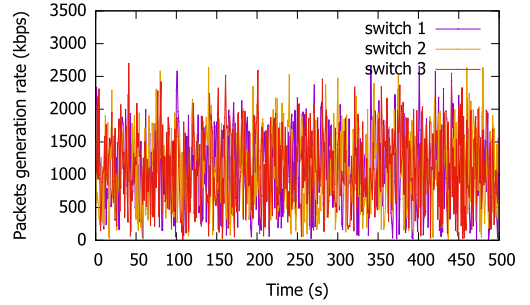


Fig. 5. Integrated traffic pattern of different switches in the EIC.

as $tlp_{k+1} = \{tlp_{k+1}^1, tlp_{k+1}^2, \dots, tlp_{k+1}^M\}$, the training data are denoted as $(x_{input}, y_{output}) = (tlp_{k+1}, L)$.

Except the new training data set with the predicted TL, the training phase of TP-DLPOCA is the same as the DLPOCA. In the dynamic channel assignment phase, the TL prediction is done before the channel assignment. Then, the central controller sends the predicted TL information tlp_{k+1} to every switch. The entire process is shown in Algorithm 2.

VI. PERFORMANCE EVALUATION

In this section, we evaluate our proposal from three aspects: 1) the prediction accuracy; 2) the performance of DLPOCA; and 3) the performance of TP-DLPOCA.

We simulate the scenarios with the configuration using C++/WILL [32] API as follows. 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 Section III-A, there are different kinds of devices

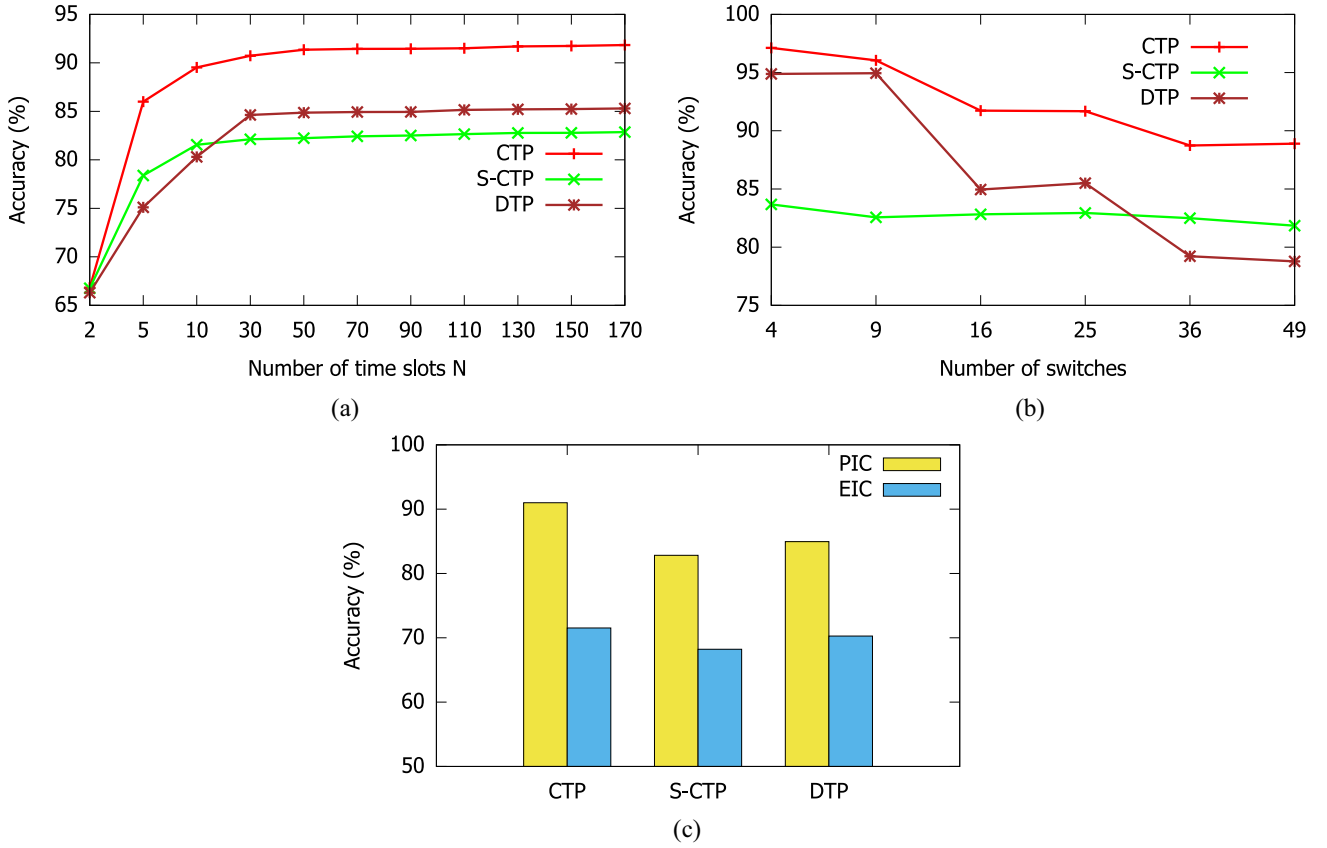


Fig. 6. Performance comparison of the three kinds of proposed mechanisms. Prediction accuracy with different (a) numbers of time slots, N , (b) numbers of switches, and (c) kinds of data generation method.

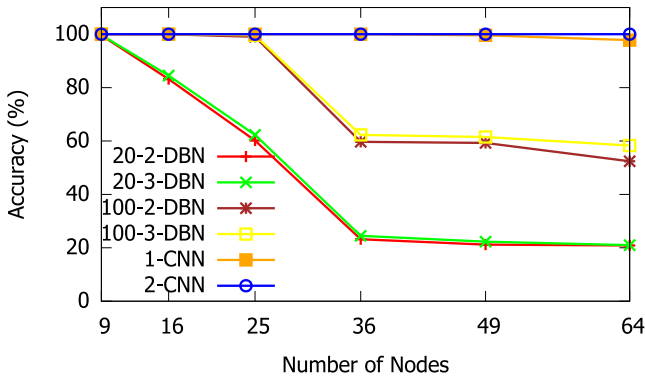


Fig. 7. Accuracy with different configuration of learning structure.

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 ten 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 this paper. 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 TL of different switches in the two different cases are shown in Table III and Figs. 4 and 5. Consider the distance between switches is randomly set ranging from 10 m 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 [14], the multichannels, multiradios 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 Section III, and the $IR(\sigma)$ is shown in Table I. The data rate of each link is set to 8 Mb/s. We conduct our network with the number of switches from 9 to 64 to show the different performances in various network environments.

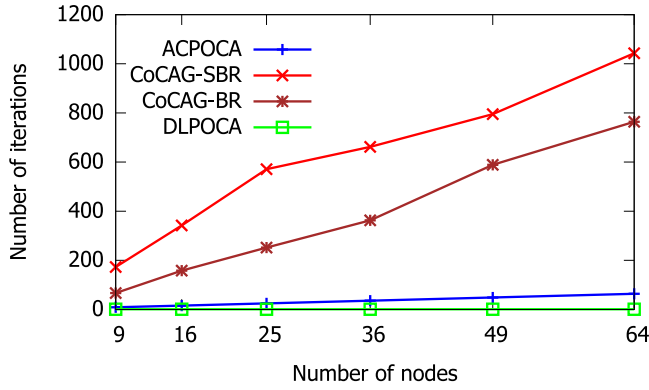


Fig. 8. Convergence compared with proposal and conventional algorithm.

A. Prediction Accuracy

At first, we evaluate the accuracy performance of TL 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 Section IV-A. In Fig. 6(a), 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. 6(b) demonstrates the accuracy of the three mechanisms with different numbers of switches. This simulation is conducted in the situation of PIC and $N = 70$ and $\Delta = 1$ s. The switches are deployed as described in Section III-A. 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

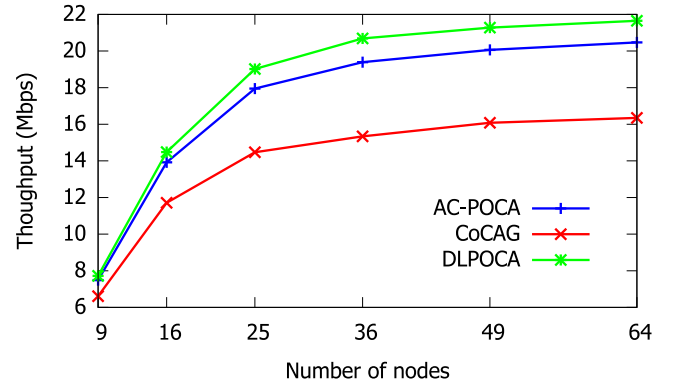


Fig. 9. Throughput compared with proposed DLPOCA and conventional algorithms.

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. 6(c), we compare the three kinds of deep learning-based TL 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 DLCA in SDN-IoT.

B. 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, and AC-POCA).

In Fig. 7, we compare the training accuracy with different learning structures, i.e., DBN with two and three 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 one and two convolution layers and two 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 and 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

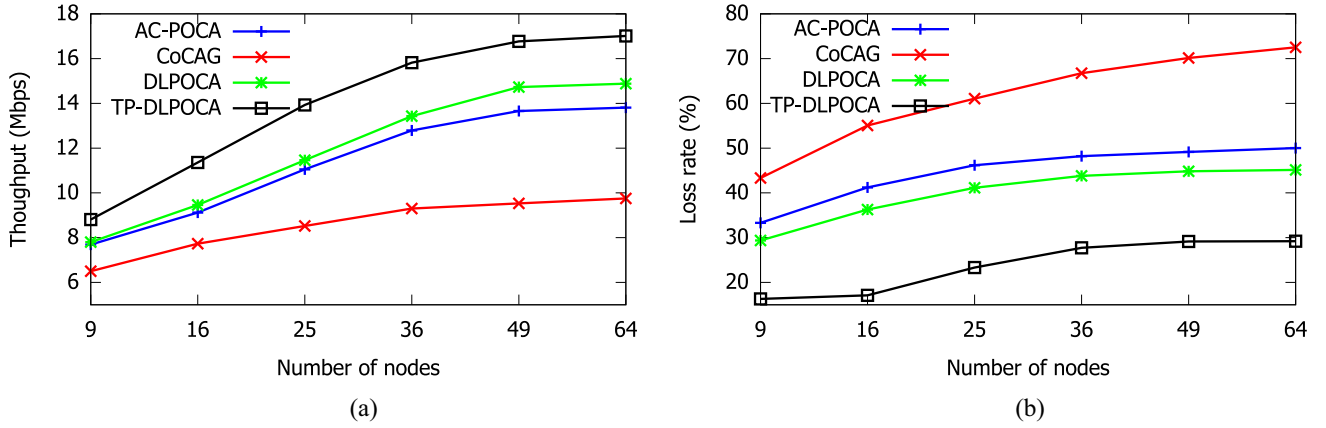


Fig. 10. Network performance comparison of the TP-DLPOCA, DLPOCA and conventional algorithms in terms of throughput, packets loss rate in the situation of PIC. (a) Throughput compared with TP-DLPOCA and conventional algorithms. (b) Packets loss rate compared with TP-DLPOCA and conventional algorithms.

accuracy result is shown in Fig. 7. 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 assignment game (CoCAG) with best response (BR) and smoothed better response (SBR) [11], and AC-POCA [14]. For ease of representation, the three methods are briefly referred to as CoCAG-BR, CoCAG-SBR, and AC-POCA, respectively. We run the channel assignment process 100 times with randomly deployment of nodes and obtain the average number of iteration times.

Fig. 8 shows the comparison result of the convergence time. As described in Section V-A, 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 convergence time and signaling overhead are significantly low.

Considering the dynamics of network, the TL and link condition may change frequently, leading to the frequent channel reassignments. Here, we consider the TL in the situation of PIC. In order to simulate such a situation, we set the frequency of channel assignment in our simulation as 10 s/1 (i.e., every 10 s execute once). During the channel assignment,

all data transmissions are paused. And then, we compare the throughput between DLPOCA, AC-POCA and CoCAG. In this simulation, the buffer of each node is set to 100 kB, 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 1 Mb/s. The packet size and signaling size both are set to 1 kB. Then, we run the simulation over 1000 s, and the throughput result is demonstrated in Fig. 9. 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.

C. Performance of the Joint Deep Learning-Based Prediction and Channel Assignment

To further improve the channel performance, we combine the intelligent DLCA with deep learning-based TL prediction. The result of Fig. 10 shows the performance of our proposed TP-DLPOCA with conventional channel assignment algorithms (i.e., CoCAG and AC-POCA). In this simulation, the parameters are considered to be the same as those in Section VI-B. Fig. 10(a) and (a) demonstrates 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

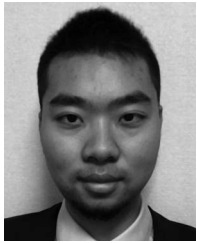
is because TP-DLPOCA can predict the congestion and give the links in the congestion area higher priority to be assigned with high quality channels. And even when the network situation changes, the proposed TP-DLPOCA can still learn from the new situation and predict the congestion to assign suitable channels. This good performance of the proposed TP-DLPOCA can be credited as the online intelligent channel assignment strategy.

VII. CONCLUSION

The explosive growth of sensing data and quick response requirements of the IoT have recently lead 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 TLs. Recently, the SDN-IoT is proposed to improve the transmission quality. Moreover, the deep learning technique has been widely researched in high computational SDN. Therefore, we first proposed a novel deep learning-based TL prediction method to predict the future TL and network congestion. Then, a DLPOCA algorithm was proposed to intelligently assign channels to each link in SDN-IoT. Finally, we combine the 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.

REFERENCES

- [1] Y. Kawamoto *et al.*, "A feedback control-based crowd dynamics management in IoT system," *IEEE Internet Things J.*, vol. 4, no. 5, pp. 1466–1476, Oct. 2017.
- [2] Y. Kawamoto *et al.*, "Effectively collecting data for the location-based authentication in Internet of Things," *IEEE Syst. J.*, vol. 11, no. 3, pp. 1403–1411, Sep. 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 Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1457–1477, 3rd Quart., 2017.
- [4] J. Ni, K. Zhang, X. Lin, and X. S. Shen, "Securing fog computing for Internet of Things applications: Challenges and solutions," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 1, pp. 601–628, 1st Quart., 2018.
- [5] S. Al-Rubaye, E. Kadhum, Q. Ni, and A. Anpalagan, "Industrial Internet of Things driven by SDN platform for smart grid resiliency," *IEEE Internet Things J.*, to be published, doi: [10.1109/JIOT.2017.2734903](https://doi.org/10.1109/JIOT.2017.2734903).
- [6] K. Sood, S. Yu, and Y. Xiang, "Software-defined wireless networking opportunities and challenges for Internet-of-Things: A review," *IEEE Internet Things J.*, vol. 3, no. 4, pp. 453–463, Aug. 2016.
- [7] Z. Qin, G. Denker, C. Giannelli, P. Bellavista, and N. Venkatasubramanian, "A software defined networking architecture for the Internet-of-Things," in *Proc. IEEE Netw. Oper. Manag. Symp. (NOMS)*, May 2014, pp. 1–9.
- [8] 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 Things J.*, vol. 3, no. 2, pp. 231–243, Apr. 2016.
- [9] 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 Things J.*, vol. 3, no. 3, pp. 354–365, Jun. 2016.
- [10] N. Zhang *et al.*, "Risk-aware cooperative spectrum access for multi-channel cognitive radio networks," *IEEE J. Sel. Areas Commun.*, vol. 32, no. 3, pp. 516–527, Mar. 2014.
- [11] 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 J. Sel. Areas Commun.*, vol. 30, no. 1, pp. 119–127, Jan. 2012.
- [12] Y. Liu, R. Venkatesan, and C. Li, "Channel assignment exploiting partially overlapping channels for wireless mesh networks," in *Proc. IEEE Glob. Telecommun. Conf. (GLOBECOM)*, Nov. 2009, pp. 1–5.
- [13] 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 *Proc. IEEE Int. Conf. Commun.*, Jun. 2009, pp. 1–6.
- [14] 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 Trans. Veh. Technol.*, vol. 67, no. 2, pp. 1672–1683, Feb. 2018.
- [15] CISCO. *The Internet of Things Reference Model White Paper*. Accessed: Mar. 2018. [Online]. Available: http://cdn.iotwf.com/resources/71/IoT_Reference_Model_White_Paper_June_4_2014.pdf
- [16] F. Al-Turjman, "Information-centric framework for the Internet of Things (IoT): Traffic modeling and optimization," *Future Gener. Comput. Syst.*, vol. 80, pp. 63–75, Mar. 2018.
- [17] K. Akkaya and M. Younis, "A survey on routing protocols for wireless sensor networks," *Ad Hoc Netw.*, vol. 3, no. 3, pp. 325–349, 2005.
- [18] S. Tozlu, M. Senel, W. Mao, and A. Keshavarzian, "Wi-Fi enabled sensors for Internet of Things: A practical approach," *IEEE Commun. Mag.*, vol. 50, no. 6, pp. 134–143, Jun. 2012.
- [19] K. Sood, S. Yu, Y. Xiang, and S. Peng, "Control layer resource management in SDN-IoT networks using multi-objective constraint," in *Proc. IEEE 11th Conf. Ind. Electron. Appl. (ICIEA)*, Jun. 2016, pp. 71–76.
- [20] M. Ojo, D. Adami, and S. Giordano, "A SDN-IoT architecture with NFV implementation," in *Proc. IEEE Globecom Workshops*, Dec. 2016, pp. 1–6.
- [21] B. Nguyen, N. Choi, M. Thottan, and J. V. Der Merwe, "SIMECA: SDN-based IoT mobile edge cloud architecture," in *Proc. IFIP/IEEE Symp. Integr. Netw. Service Manag. (IM)*, May 2017, pp. 503–509.
- [22] H. Skalli, S. Ghosh, S. K. Das, and L. Lenzini, "Channel assignment strategies for multiradio wireless mesh networks: Issues and solutions," *IEEE Commun. Mag.*, vol. 45, no. 11, pp. 86–95, Nov. 2007.
- [23] P. B. F. Duarte, Z. M. Fadlullah, K. Hashimoto, and N. Kato, "Partially overlapped channel assignment on wireless mesh network backbone," in *Proc. IEEE Glob. Telecommun. Conf. (GLOBECOM)*, Miami, FL, USA, Dec. 2010, pp. 1–5.
- [24] N. Kato *et al.*, "The deep learning vision for heterogeneous network traffic control: Proposal, challenges, and future perspective," *IEEE Wireless Commun.*, vol. 24, no. 3, pp. 146–153, Jun. 2017.
- [25] Z. Fadlullah *et al.*, "State-of-the-art deep learning: Evolving machine intelligence toward tomorrow's intelligent network traffic control systems," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2432–2455, 4th Quart., 2017.
- [26] B. Mao *et al.*, "Routing or computing? The paradigm shift towards intelligent computer network packet transmission based on deep learning," *IEEE Trans. Comput.*, vol. 66, no. 11, pp. 1946–1960, Nov. 2017.
- [27] F. Tang *et al.*, "On removing routing protocol from future wireless networks: A real-time deep learning approach for intelligent traffic control," *IEEE Wireless Commun.*, vol. 25, no. 1, pp. 154–160, Feb. 2018.
- [28] M. A. Hoque, X. Hong, and F. Afroz, "Multiple radio channel assignment utilizing partially overlapped channels," in *Proc. IEEE Glob. Telecommun. Conf. (GLOBECOM)*, Honolulu, HI, USA, Nov. 2009, pp. 1–7.
- [29] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017, doi: [10.1109/TPAMI.2016.2577031](https://doi.org/10.1109/TPAMI.2016.2577031).
- [30] Y. Kim, "Convolutional neural networks for sentence classification," in *Proc. Conf. Empirical Methods Nat. Lang. Process. (EMNLP)*, 2014, pp. 1746–1751.
- [31] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3D convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 4489–4497.
- [32] WILL API. Accessed: Dec. 2017. [Online]. Available: <https://scarsty.gitbooks.io/will/content/>



Fengxiao Tang (S'16) received the B.E. degree in measurement and control technology and instrument from the Wuhan University of Technology, Wuhan, China, in 2012, and the M.S. degree in software engineering from Central South University, Changsha, China, in 2015. He is currently pursuing the Ph.D. degree at the Graduate School of Information Sciences, Tohoku University, Sendai, Japan.

His current research interests include unmanned aerial vehicles system, game theory optimization,

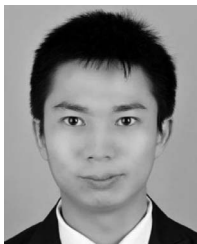
and deep learning.



Zubair Md. Fadlullah (M'11–SM'13) is an Associate Professor with the Graduate School of Information Sciences, Tohoku University, Sendai, Japan. His current research interests include 5G, social networks, smart grid, and network security.

Mr. Fadlullah was a recipient of the prestigious Dean's and President's Awards from Tohoku University for his outstanding research contributions in 2011, the IEEE Communications Society Asia-Pacific Board Outstanding Young Researcher Award in 2015, and the NEC Tokin Foundation Award in 2016.

Award in 2015, and the



Bomin Mao (S'15) received the B.Sc. degree in telecommunications engineering and M.S. degree in electronics and telecommunications engineering from Xidian University, Xi'an, China, in 2012 and 2015, respectively. He is currently pursuing the Ph.D. degree at the Graduate School of Information Sciences, Tohoku University, Sendai, Japan.

His current research interests include wireless networks, software-defined networks, and quality of service, particularly with applications of machine intelligence and deep learning.



Nei Kato (M'03–A'04–SM'05–F'13) is a Full Professor and the Director of Research Organization of Electrical Communication, Tohoku University, Sendai, Japan. He has been engaged in research on computer networking, wireless mobile communications, satellite communications, ad hoc and sensor and mesh networks, smart grids, Internet of Things, big data, and pattern recognition. He has authored or co-authored over 400 papers in prestigious peer-reviewed journals and conferences.

Mr. Kato was a recipient of the Minoru Ishida Foundation Research Encouragement Prize in 2003, the Distinguished Contributions to Satellite Communications Award from the IEEE Communications Society, Satellite and Space Communications Technical Committee in 2005, the FUNAI Information Science Award in 2007, the TELCOM System Technology Award from the Foundation for Electrical Communications Diffusion in 2008, the IEICE Network System Research Award in 2009, the IEICE Satellite Communications Research Award in 2011, the KDDI Foundation Excellent Research Award in 2012, the IEICE Communications Society Distinguished Service Award in 2012, the IEICE Communications Society Best Paper Award in 2012, the Distinguished Contributions to Disaster-Resilient Networks R&D Award from the Ministry of Internal Affairs and Communications, Japan, in 2014, the Outstanding Service and Leadership Recognition Award from the IEEE Communications Society Ad Hoc and Sensor Networks Technical Committee in 2016, the Radio Achievements Award from the Ministry of Internal Affairs and Communications, Japan, in 2016, and the Best Paper Award from IEEE ICC/GLOBECOM/WCNC/VTC. He has been the Vice-President (Member and Global Activities) of the IEEE Communications Society since 2018, has been the Editor-in-Chief of the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY since 2017, was the Editor-in-Chief of the *IEEE Network Magazine* from 2015 to 2017, has been the Associate Editor-in-Chief of the IEEE INTERNET OF THINGS JOURNAL since 2013, and is the Chair of the IEEE Communications Society Sendai Chapter. He served as a Member-at-Large on the Board of Governors, IEEE Communications Society from 2014 to 2016, the Vice Chair of the Fellow Committee of the IEEE Computer Society in 2016, and a member of the IEEE Computer Society Award Committee from 2015 to 2016 and the IEEE Communications Society Award Committee from 2015 to 2017. He has also served as the Chair of the Satellite and Space Communications Technical Committee from 2010 to 2012 and the Ad Hoc and Sensor Networks Technical Committee from 2014 to 2015 of the IEEE Communications Society. He was a Distinguished Lecturer of the IEEE Communications Society and the IEEE Vehicular Technology Society. He is a Fellow of the IEICE.