Reliable and Scalable Routing under Hybrid SDVN Architecture: A Graph Learning based Method

Zhuhui Li, Liang Zhao, *Member, IEEE,* Geyong Min, Ahmed Al-Dubai, *Senior Member, IEEE,* Ammar Hawbani, Albert Y. Zomaya, *Fellow, IEEE*, and Chunbo Luo

***Abstract*—Greedy routing efficiently achieves routing solutions for vehicular networks due to its simplicity and reliability. However, the existing greedy routing schemes have mainly considered simple routing metrics only, e.g., distance based on the local view of an individual vehicle. This consideration is insufficient for analysing dynamic and complicated vehicular communication scenarios. This shortcoming inevitably degrades the overall routing performance. Software-Defined Vehicular Network (SDVN) and Graph Convolutional Network (GCN) could break these limitations. Thus, this paper presents a novel GCN-based greedy routing algorithm (NGGRA) in the hybrid SDVN. The SDVN control plane trains the GCN decision model based on the globally collected data. The vehicle with transmission requirements can adopt this model for inferring and making the routing decision. The new proposed node-importance-based graph convolutional network (NiGCN) model analyses multiple metrics in the dynamic vehicular network scenario. Meanwhile, SDVN architecture offers a global view for model training. Extensive simulation results demonstrate that NiGCN outperforms most popular GCN models in training efficiency and accuracy. In addition, NGGRA can improve the packet delivery ratio and latency substantially compared with its counterparts.**

***Index Terms*—Software-defined vehicular network, vehicular ad-hoc networks, routing, deep learning, graph convolutional networks.**

I. Introduction

Vehicular ad-hoc network (VANET) plays an essential role in enabling a myriad of applications, such as traffic control, collision avoidance, and emergency information transportation in intelligent transportation systems (ITS) [1]. Generally, VANETs consist of vehicles equipped with onboard units (OBU) together with GPS, base stations (BS), and roadside units (RSU) on the segment. The vehicle-to-vehicle (V2V) communications enabled by OBUs, as well as vehicle-to-infrastructure (V2I) communication between OBUs and RSUs, are the primary methods in VANETs [2]. The packets need to be relayed among vehicles and RSUs through a multi-hop approach. It is the foundation for all communications in VANETs. Moreover, routing is the method to find this multi-hop path. Reliability and efficiency are critical factors for evaluating the routing schemes since massive safety-related information could be transmitted in VANETs. Meanwhile, the highly dynamic traffic topology makes pursuing these two factors simultaneously a challenging issue.

The greedy routing scheme, as a promising paradigm in VANET, have its natural advantage in routing efficiency since it needs no additional information exchange during the routing discovery. Regularly exchanged beacons maintain the neighbour information, and the vehicle can make relay decisions based on this local information. Compared to the proactive routing scheme, the routing discovery efficiency and overhead can be reduced [3]. The greedy routing can also adapt to the dynamic change of the network topology according to the exchanged beacons. However, the greedy routing scheme cannot satisfy the reliability requirement, and the performance is unstable in urban scenarios [4]. In some greedy algorithms, the vehicle only utilises simple metrics (such as distance) to decide its relay vehicle, ignoring the complex urban scenario with various obstacles and constantly changing network topology [5]–[7]. Meanwhile, most greedy routing schemes make relay decisions based on the vehicles' one-hop view, which inevitably leads to the optimal maximum, the inherent disadvantage of greedy algorithms [6], [8]. Some improved greedy routing schemes introduce additional external information out of the vehicles to improve the reliability of the routing performance, such as digital maps, bus routes, junction information etc. [9], [10]. This additional information assists the routing scheme in accurately judging and predicting possible future traffic. However, the possible deterioration of computation efficiency and overhead must be seriously considered.

For the shortcomings of the current greedy algorithms, we introduce the Software-Defined Vehicular Network (SDVN) and graph convolutional network (GCN) to achieve the tradeoff between efficiency and reliability. GCN is one of the most suitable models for analysing the vehicular network since it can naturally be represented as a graph. GCNs define convolution and readout operations on irregular graph structures to capture common local and global structural patterns [11]. Thus, this model can explore relationships and possible routing paths among the neighbouring vehicles based on their multiple features, including position, velocity, acceleration, and vehicle load. To make the GCN model

Zhuhui Li and Liang Zhao are with the School of Computer Science, Shenyang Aerospace University, Shenyang, China. E-mail: zhuhui.li7217@gmail.com, lzhao@sau.edu.cn. Liang Zhao is the corresponding author.

Geyong Min and Chunbo Luo are with the Department of Computer Science, University of Exeter, UK. E-mail: g.min@exeter.ac.uk, C.Luo@exeter.ac.uk.

Ahmed Al-Dubai is with the School of Computing, Edinburgh Napier University, UK. E-mail: a.al-dubai@napier.ac.uk.

Ammar Hawbani is with the School of Computer Science and Technology, University of Science and Technology of China, Hefei, China. E-mail: anmande@ustc.edu.cn.

Albert Y. Zomaya is with the School of Computer Science, University of Sydney, Australia. E-mail: albert.zomaya@sydney.edu.au.

adaptive to vehicular network routing, we further consider the node importance and propose a novel GCN model, namely, the node-importance-based graph convolutional network (NiGCN). Node importance can be seen as the impact of one node on its surrounding nodes and the whole network. This model can improve the training accuracy and efficiency compared with the existing popular GCN models.

With regular one-hop beacon exchange between vehicles, NiGCN helps vehicles better analyse and predict the status of themselves and their multi-hop neighbours, which is helpful to enlarger one vehicle's vision and analyse capability with simple liner computation requirements. However, the training for the NiGCN model is difficult for an individual vehicle due to its limited computation resource and scarce training data. Thus, we want to introduce SDVN architecture. The separation of the data plane and the control plane is the core idea of SDVNs [12], [13]. Since the control plane in SDVNs collects the information of vehicles periodically, it senses the vehicular network globally [14]. Typically, the control plane computes all requested routing paths from the data plane centralised in SDVN architectures. These centralised management methods inevitably increase the communication and computation overhead, and high mobility of vehicles and possible packet loss make the connectivity between control and data plane unstable [15]. Thus, we ease the communication and computation burden for the control plane by assigning the control plane with the decision model training task, and the data plane (vehicles) makes the relay decision independently based on the centralised trained decision model. This hybrid SDVN architecture significantly reduces the communication between the control and data plane. Single point failure can be avoided, and stability and scalability of the network management can be well improved.

To this end, based on the NiGCN model and hybrid SDVN architecture, we propose a novel routing algorithm, namely, a novel NiGCN-based greedy routing algorithm (NGGRA) for hybrid SDVNs. As a premise, all vehicles can exchange their features with their neighbours. Once one node transmits the packets, this vehicle and its neighbours can utilise a centralised trained decision model inputted with exchanged features of neighbour vehicles. This model can select the relay node among the neighbours can be selected, and the source vehicle will transmit packets to it. This next-hop vehicle selection step will be repeated and iterated continuously, allowing all packets successfully transmitted to the destination vehicle. Additionally, the controller periodically collects the related routing information as the training data for these models. Table I states the comparison of NGGRA with other greedy and SDVN routing algorithms. The significant contributions of this paper are summarised as follows.

- A novel hybrid SDVN architecture is proposed to assist the routing computation. The global view of SDVN can be well utilised. Meanwhile, this architecture significantly saves computing resources and reduces network overhead for both the controller and vehicles while guaranteeing high Quality-of-Service (QoS).
- A novel graph neural network model is designed, namely node-importance based graph convolutional network (NiGCN). The node importance is integrated into the training model. This training method achieves higher accuracy by differing the influences each node exerts on the neighbouring nodes and maintains the simplicity of the GCN model.
- A SDVN NiGCN-based routing algorithm, NGGRA, is developed. NGGRA takes advantage of the strong learning capacity of the NiGCN model and fully considers various networking-related features and the interconnections among them. Meanwhile, under the hybrid SDVN architecture, the proposed routing algorithm guarantees the global view while reducing the network overhead.

The rest of the paper is organised as follows. Section II introduces the hybrid SDVN architecture and presents problem formulation. Our novel NiGCN model is described in detail in Section III. Section IV presents our routing algorithm architecture, NGGRA. In Section V, the simulation results on NiGCN and NGGRA are presented and analysed from different perspectives. Finally, Finally, the conclusion is drawn out in Section VI.

II. Network Model and Problem Formulation

Section II.A. describes the network model. The problem formulation of this work and the solution will be presented in Section II.B.

A. Hybrid SDVN

In the traditional SDVNs, the control plane collects the information of vehicles periodically to sense the vehicular network globally [14] as assistance for routing decisions. However, the centralised architecture is vulnerable to the single-point failure [17]. It will also increase network overhead due to the frequent information exchange between data and the control plane, which is an inherent disadvantage compared to the distributed architecture [15].

Thus, in this section, we present the hybrid SDVN architecture in urban scenarios, as shown in Fig.1. We only make innovations in the communication logic between the control plane and the data plane compared with traditional SDVN architecture, while the physical configuration remains the same. A logically centralised software defined network (SDN) control plane copes with modern vehicular networks' distributed and heterogeneous nature. Multiple controllers could cooperate to realise all functions to support the data plane in real life. The control plane is not responsible for the actual forwarding rules computation in our architecture. It mainly trains the routing decision model based on the massive routing-related data updated from the data plane. The details of the decision model will be introduced in Section III and Section IV. The control plane connects all BSs and RSUs within its coverage area. RSUs serve the vehicles within their coverage area, while the BS serves the vehicles out of the coverage area of RSUs. RSUs and BS mainly collect routing-related information from the vehicle and update them to the control plane. Meanwhile, once the decision model trained by the control plane is updated, they will distribute them

TABLE I
COMPARISON ON THE GREEDY ROUTING ALGORITHM IN VEHICULAR NETWORK

Algorithm	Efficiency	Reliability	Overhead	Scalability	Metric used in routing	Additional information ?	Vehicle's view range	Advantage	disadvantage
GPSR [5]	High	Low	Low	High	Vehicle Location	No	One Hop	Simple and efficient; Low overhead during the beacon exchange	The urban scenario with obstacles have not been seriously considered.
GyTAR [16]	Medium	Medium	Medium	Low	Traffic Density, Curvemetric Distance	Digital Map	Neighbouring Junction	It is a junction-based algorithm, so the obstacle can be avoided	Segment with low vehicle density have not been considered.
HRLB [9]	Low	Medium	High	Low	Vehicle Density, Historic Routing Information, Vehicle Location, Transmission Load	Global information achieved from the control plane, Digital Map	Global View from Control Plane	The global view of SDVN assist routing discovery	Overhead between control and data plane is large. No effective methods to avoid local optimum.
QGrid [10]	Medium	Medium	Medium	Low	Vehicle Location, Vehicle Type, Historic Routing Information	Digital Map, RL Decision Model. Bus Information	One Hop	Powerful RL model trained from historic routing information assists routing discovery.	Additional information such as the digital map and bus trace is needed. The existence of bus give some extra constraints on some rural scenarios
FLGR [6]	High	Low	Medium	High	Vehicle Location, Velocity	Fuzzy Logic	One Hop	Multi metrics are analysed with fuzzy logic. The neighbour statues can be well evaluated.	Limited view and obstacles in the urban scenario are not considered.
NGGRA	High	High	Medium	Medium	Vehicle Location, Velocity, Transmission Load	NiGCN Decision Model	Global view from control plane	Powerful NiGCN model and SDVN assist routing discovery	Extra overhead are introduced for the feature exchange

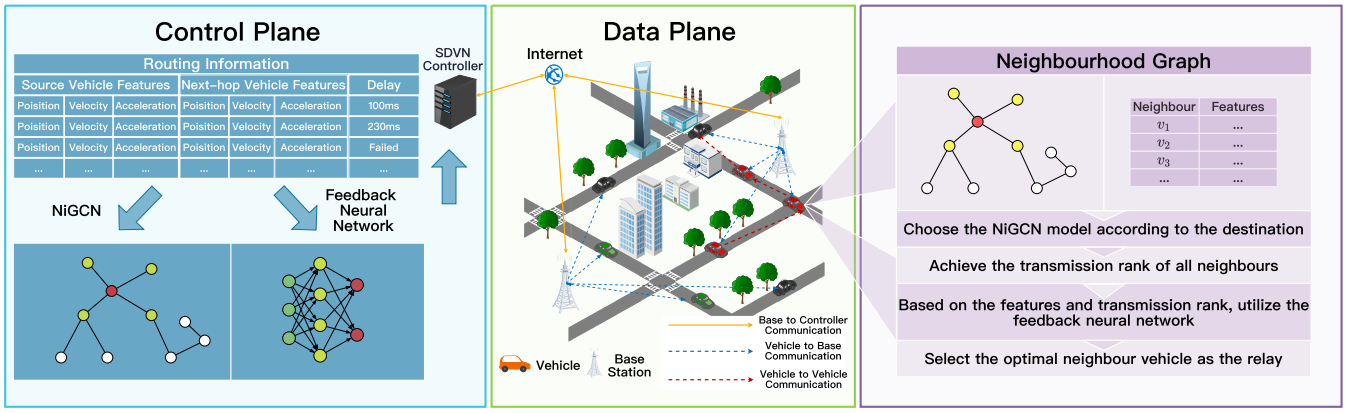

Fig. 1. A typical urban scenario of NGGRA (one vehicle requesting for transmitting traffic information to another vehicle).

to all vehicles. Once one vehicle needs to transmit packets, the source and neighbour vehicles will utilise the decision model to decide the next-hop relay vehicle. Vehicles will repeat this process iteratively and greedily until the packets reach the destination vehicle. The details of this routing process will be introduced in detail in Section IV. Under this architecture, the hybrid SDVN allows for more flexibility and scalability of the vehicular network. This architecture can avoid single-point failure because the control plane is not directly involved in routing. Even if the network between the data and the control plane is unstable, it will not affect the data transmission among the data planes. In addition, the rapidly growing data plane data volume will not bring too much computational burden to the control plane. Unlike real-time computing routing decisions, the training of decision-making models does not have strict real-time requirements. It greatly improves the scalability of the architecture. Moreover, The limited vision issues can be satisfactorily solved with a reasonable routing management method between the control plane and data plane. At the same time, the network overhead can be well reduced.

Each vehicle is equipped with both the IEEE 8021.11p and LTE interfaces as this hybrid architecture has been widely adopted. V2V communications use DSRC based on the IEEE 802.11p protocol. Meanwhile, V2I communication utilises LTE. The communications among RSUs, BS, and controllers are through wired connections to minimise the probability of unsuccessful transmission and packet loss.

B. Problem Formulation

The goal of the routing algorithm is to successfully transmit a given number of packets from the source vehicle src to the destination node des with good transmission quality. We aim to maximise the delivery ratio and minimise the average transmission delay. First, we will present specific notations of the relevant metrics and concepts.

1) Notation: First, our optimized variable is the routing path $P_i(v_0, v_1, \ldots, v_n)$ for each packet pkt_i. v_0 and v_n are the source node src and destination node des, respectively. The delivery ratio r and the average delivery delay T

are two mains metrics to evaluate the transmission quality comprehensively.

For the first metric, we consider the delivery ratio r as the ratio of packet set pkt_set_m that successfully arrived at their specific destination nodes among the entire packet set pkt_set_n:

$$r = \frac{sizeof(pkt\_set_m)}{sizeof(pkt\_set_n)} \tag{1}$$

The delivery delay T is the average delivery delay of successful packet transmissions of pkt_set_m, which can be formulated as:

$$T = \frac{\sum T(P_i)}{sizeof(pkt\_set_m)}, \forall P_i,\ i \in pkt\_set_m \tag{2}$$

For each packet pkt_i, its delivery delay, $T(P_i)$, is the sum of four items:

$$T(P_i) = \sum_{k=0}^{n-1}(t_{trans|pkt_i}^{v_k,v_{k+1}} + t_{prop}^{v_k,v_{k+1}} + t_{proc}^{v_k,v_{k+1}} + t_q^{v_k,v_{k+1}}) \quad \forall v_k \in P_i, \forall P_i, i \in pkt\_set_m \tag{3}$$

where $t_{trans|pkt_i}^{v_k,v_{k+1}}$ is the transmission delay in sending a packet pkt_i within a single wireless hop from the vehicle v_k to the v_{k+1}. $t_{prop}^{v_k,v_{k+1}}$ is the propagation delay which depends mainly on the distance. $t_{proc}^{v_k,v_{k+1}}$ and $t_q^{v_k,v_{k+1}}$ are the processing delay and queuing delay, respectively. Both of them are highly dependent on the vehicle's current state, where the excessive load may even cause serious problems such as longer delay or packet loss. These four metrics have all been formulated by the standard equations in [18] [19].

Based on the components of $T(P_i)$, the vehicle load, number of hops, and distance of a routing path significantly affect the packet delivery delay. If the status of each vehicle on P_i is well sensed and evaluated, $T(P_i)$ will be significantly reduced, and the packet loss will be well avoided. Thus, $T(P_i)$ directly reflects the network architecture and routing algorithm's perception and analysis capabilities. SDVN routing scheme always contains another part of delay: vehicle-controller communication time t_{veh_con}. It is the time cost of the source vehicle spending on communicating with the control plane to obtain routing path information. However, NGGRA, as an SDVN routing algorithm, does not need this part since the vehicle makes the routing decision independently without direct communication with the controller plane. The communication between data and the control plane is separated from the routing process.

2) Objective Function: Our objective is to improve the quality of the computed routing path, which means that the networking performance in terms of delivery ratio and the delivery delay should be optimal to ensure high-quality data transmission. Thus, the objective function can be formulated as follows.

main objective

$$\min\ T \tag{4}$$

$$\max\ r \tag{5}$$

s.t.

$$\sum_{i=1}^{n}\lambda_{v_i,v_j}^{P_h} - \sum_{j=1}^{n}\lambda_{v_j,v_i}^{P_h} = \begin{cases} 1 & v_i = src \\ -1 & v_i = des \\ 0 & v_i \neq src, des \end{cases} \quad v_i, v_j \in P_h, \forall P_h, h \in pkt\_set_m \tag{6}$$

$$\lambda_{v_i,v_j}^{P_h} \geq 0, \quad (v_i, v_j) \in P_h, \quad \forall P_h, h \in pkt\_set_m \tag{7}$$

$$\lambda_{v_i,v_j}^{P_h} + \lambda_{v_j,v_i}^{P_h} \leq 1, \quad (v_i, v_j) \in P_h, \quad \forall P_h, h \in pkt\_set_m \tag{8}$$

$$d_{v_i,v_{i+1}} \leq d_{trans}^{max}, \quad v_i \in P_h, \quad \forall P_h, h \in pkt\_set_m \tag{9}$$

$$\omega_h^{v_i,v_j} \leq \gamma_{v_j}, \quad (v_i, v_j) \in V, \quad \forall P_h, h \in pkt\_set_m \tag{10}$$

First, the objective of our routing algorithm is divided into two parts: maximising delivery ratio and minimising transmission delay. However, implementing these two objectives would lead to the overall optimal performance to harmonise the delivery ratio and delay. If we model and predict the network well through a well-trained powerful decision model, it is possible to pursue a high delivery ratio and low delay simultaneously. In the constraints, $\lambda_{v_j,v_i}^{P_h}$ means the link state between vehicle v_i and vehicle v_j (6). It is a boolean value. If v_i needs to transmit packets to v_j during the forwarding process of P_h, this value will be true. Otherwise, it is false. (7) and (8) are based on the flow conservation concept, which is responsible for routing computation for all packets. (8) is utilised to avoid the loops in the routing computation. In the next constraint (9), $dis_{v_i,v_{i+1}}$ means the distance between v_i and v_{i+1}. dis_{trans}^{max} denotes the maximum transmission range. We need to ensure the distance of each adjacent vehicle pair in P_h within the maximum transmission range. Otherwise, the packet loss is bound to happen. The last constraint (10) is related to capacity. $\omega_h^{v_i,v_j}$ represents the size of packet h in which this packet is transmitted from v_i to v_j. γ_{v_j} means the remaining capacity of v_j. If the queue of the vehicle is full, it will not be considered during the routing computation until the queue of this vehicle becomes spare.

3) Solutions on problem: : Since each vehicle constantly moves, it is difficult to satisfy these two objectives. To simplify this complex and chaotic vehicle scenario, we assume that each vehicle v_i knows the minimum delay for delivering the packet to the destination vehicle v_d at time t. To cover two objectives, we express this metric as follows:

$$\tau_{v_i}(v_d, t) = \begin{cases} \min T(v_i, v_d), & \text{if } P(v_i, \ldots, v_d) \text{ exists at } t \\ \infty, & \text{otherwise} \end{cases} \tag{11}$$

Meanwhile, we assume that each node v_i is aware of its neighbor set $N_i = v_1, ..v_m$, and know the one-hop transmission delay at time t. We also consider link failure in this metric, which is expressed as follows:

$$\sigma_{v_i}(v_j, t) = \begin{cases} T(v_i, v_j), & \text{if } \text{link}(v_i, v_j) \text{ is available at } t \\ \infty, & \text{otherwise} \end{cases} \tag{12}$$

With these two metrics, optimal routing can be easily achieved.

Lemma 1: There is a packet $pkt\,(s,d)$ generated from vehicle v_s, and its destination vehicle is v_d. Each node v_i holding the packet selects v_{i+1} among its neighbor set N_p as the relay vehicle and forward packet to v_{i+1}, which v_{i+1} satisfied:

$$v_{i+1} = \underset{v_j \in N_p}{\operatorname{argmin}} \sigma_{v_i}\left(v_j, t\right) + \tau_{v_j}\left(v_d, t + \sigma_{v_i}\left(v_j, t\right)\right) \quad (13)$$

If this process is repeated iteratively utill this packet reaches destination v_d, the series consist of vehicles who have held the packet $VS\left(v_1, v_2, \ldots.., v_n\right)$ is the optimal routing path which satisfy two main objectives (4) and (5). In VS, $v_1 = v_s$, $v_2 = v_d$.

Proof 1 (Proof of Lemma 1):

Base case: if v_1 selects another neighbor v_2' as the relay node instead of optimal relay node v_2. And v_2' is included in the optimal routing path where the delay is minimized; Then the inequality can be achieved according to (14)

$$\begin{aligned} &\sigma_{v_1}\left(v_2', t\right) + \tau_{v_2'}\left(v_d, t + \sigma_{v_1}\left(v_2', t\right)\right) < \\ &\sigma_{v_1}\left(v_2, t\right) + \tau_{v_2}\left(v_d, t + \sigma_{v_1}\left(v_2, t\right)\right) = \tau_{v_1}(v_d, t) \end{aligned} \quad (14)$$

which is contrary to that $\tau_{v_1}(v_d, t)$ is the minimized delay. Thus, v_1 and v_2 must be included in optimal routing path.

Inductive step: When the packet reaches v_k, the corresponding delay can be expressed as:

$$\mathrm{T}\left(v_1, v_k\right) = \sum_{i=1}^{k-1} T\left(v_i, v_{i+1}\right) = \tau_{v_1}\left(v_d, t\right) - \tau_{v_k}\left(v_d, t + T\left(v_1, v_k\right)\right) \quad (15)$$

If v_k choose another neighbor v_{k+1}' as relay in stead of v_{k+1} to achieve the minimized delay. Then

$$\begin{aligned} &T\left(v_1, v_k\right) + \sigma_{v_k}\left(v_{k+1}', t\right) + \tau_{v_{k+1}'}\left(v_d, t + \sigma_{v_k}\left(v_{k+1}', t\right)\right) < \\ &\mathrm{T}\left(v_1, v_k\right) + \sigma_{v_k}\left(v_{k+1}, t\right) + \tau_{v_{k+1}}\left(v_d, t + \sigma_{v_k}\left(v_{k+1}, t\right)\right) = \tau_{v_1}(v_d, t) \end{aligned} \quad (16)$$

which is also contrary to that $\tau_{v_1}(v_d, t)$ is the minimized delay. So, v_{k+1} should be the member of optimal routing path.

To this end, we have proved Lemma 1.

The key components of this forwarding solution are $\tau_{v_i}\left(v_d, t\right)$ and $\sigma_{v_i}\left(v_j, t\right)$. However, it is challenging to accurately achieve these two functions in the real vehicular network due to the complex and dynamic network topology. Complicated urban terrain and limited view for one vehicle also bring huge difficulty to calculating $\tau_{v_i}\left(v_d, t\right)$ precisely. Thus, we need to use models to predict and estimate these two functions as accurately as possible. The GCN is an excellent option for the estimation here.

III. NIGCN

Deep learning allows computational models to discover intricate structures in large-scale datasets. It adopts the back-propagation algorithm to find out how a machine should change its internal parameters to compute the representation in each layer from the representation in the previous layer [20]–[22]. Therefore, deep learning has become the most suitable method to identify the patterns in a complex and dynamic vehicular network with massive data generations and transmissions. In addition, since the vehicular network can be presented as a graph, GCN could be the most appropriate deep learning model for this perspective. Meanwhile, the node importance could bring more learning capacity to the GCN model. The importance of different nodes in the network varies widely, and the influence they radiate is also different. Different influences will inevitably affect the classification of the neighbour nodes and the entire network. Thus, this section will introduce NiGCN in detail. We present the node importance strategy to construct an arbitrary NiGCN and directly outline its benefits in the vehicular network.

A. GCN

At first, we briefly introduce GCN [23]. This work defines the convolution operation on the graph. Each convolutional layer in GCNs updates the feature vector representation of each node in the graph by integrating the features of its neighbouring nodes. To be specific, the layer-wise forward-propagation operation of GCNs can be expressed as

$$H^{l+1} = \sigma(\widetilde{D}^{-\frac{1}{2}}\widetilde{A}\widetilde{D}^{-\frac{1}{2}}H^{l}W^{l}) \quad (17)$$

in which H^l is the vector representations of all nodes in this graph presented as the input of this current graph convolutional layer. $\widetilde{A} = A + I_N$ is used to integrate the feature vectors of neighbour nodes and itself, where A is the adjacency matrix of the graph, and I_N is the identity matrix. A can integrate the neighbours' feature, and the introduction of I_N ensures that this model considers the old features of each node itself during the node representations updating. $\widetilde{D}^{-\frac{1}{2}}$ is the diagonal node degree matrix. This matrix normalises $\widetilde{A}$ so that the scale of each node's feature vectors remains the same. This is vital for maintaining the stability of the model. W^l is a trainable weight matrix representing a linear transformation for the dimension change of the feature. The dimension of W^l depends on the dimensions of the input and output, i.e., the number of columns in H^l and H^{l+1}, respectively. $\sigma(.)$ denotes an activation function like ReLU.

Introducing the diagonal node degree matrix $\widetilde{D}^{-\frac{1}{2}}$ makes all nodes even to each other. However, different nodes in the graph must have different importance. They will exert different degrees of influence on their neighbours. A high-status node will influence its connected neighbours more than a weaker node. Treating all nodes equally would not reflect this distinction. Thus, we need to find a method to sense the different influences of different nodes while maintaining stability in the model simultaneously.

Recent work started by considering the connection between different pairs of neighbours. In a graph attention network (GAT) [18], the node importance has been considered based on the features of nodes and their neighbours. However, the adjacency matrix has not been thoroughly utilised in GAT. The graph structure can also contribute to the node importance except for the features of nodes. This information can be more determinant during the training. Thus, we need

to explore more node-importance-related information from the graph structure.

To this end, we propose a method which utilises the degree matrix to construct an arbitrary NiGCN. First, we find the degree of each node in the network by analysing the corresponding graph. Then, we concatenate this degree information to the features of each node. Finally, we utilise the traditional graph convolutional layer.

$$H^{1,n} = \sigma(\widetilde{D}^{-\frac{1}{2}}\widetilde{A}\widetilde{D}^{-\frac{1}{2}}(H^0||\widetilde{D})W_n^0) \tag{18}$$

As shown in Eq. (18), the original features H^0 will be concatenated with the degree matrix $\widetilde{D}$, and the new feature $H^0||\widetilde{D}$ will play the role of input. During the process, $H^0||\widetilde{D}$ is multiplied with the trainable weights W_l and transforms into higher-level features; Thus, the degree matrix representing the node importance can be learnt through the model to give the model the sensibility on different node importance. Then the neighbour information integration will be processed by symmetric normalized Laplacian matrix $\widetilde{D}^{-\frac{1}{2}}\widetilde{A}\widetilde{D}^{-\frac{1}{2}}$. At last, the activation function $\sigma(.)$ is also utilized. This method integrates the degree of the node into the transformation to the higher-level representations, and the node importance-related information has been naturally incarnated in the data flow during training.

To stabilize the learning process, we utilize the multi node-importance mechanism similar to [18], [19]. In particular, K independent node importance strategies execute the transformation of Eq. (18), and then K high-level representations $H^{1,1}, H^{1,2}, ..., H^{1,n}, ..., H^{1,K}$ can be achieved. Their features are concatenated, resulting in the following output feature representation:

$$H^1 = ||_{n=1}^{K} H^{1,n} \tag{19}$$

where $||$ represents the concatenation. And $H^{1,n}$ means the n-th higher-level feature processed by the node importance strategy. After the simulation, we set K as 3 to achieve the best performance. The concatenated features H^1 will be utilised as the input and processed by a traditional GCN layer as shown in Eq. (20),

$$L = \sigma(\widetilde{D}^{-\frac{1}{2}}\widetilde{A}\widetilde{D}^{-\frac{1}{2}}H^1W^1) \tag{20}$$

We can obtain the result L after the computation of this layer. The comparison between NiGCN and other popular GCN models will be presented in Section V. The node importance strategy and multi-node importance are illustrated in Fig. 2.

In different scenarios, the degree could have different meanings. In the citation network like Cora, Citeseer, and Pubmed, a high degree represents a higher number of citations, which means that this paper has a high status in its corresponding field. It is also more likely that the paper with a citation relationship shares the same field. In the VANET scenario, a high degree means a higher opportunity to forward data packets. With more neighbours, the probability of finding the most suitable relay node is greater. Thus, during this routing process, if the routing scheme selects a neighbour with more degrees as the next hop, the routing selection achieves more options and increases the probability of successfully transmitting packets.

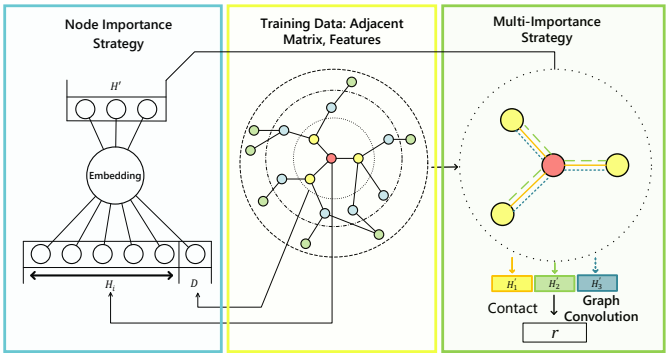

Fig. 2. An illustration of NiGCN.

On the other hand, a higher degree can also represent a high transmission load. However, the training for the NiGCN is entirely subjective, without any prejudice. The construction of NiGCN is based on the training data. Our model is able to learn the tradeoff between forwarding opportunities and transmission load during the training. With sufficient data, NiGCN can accurately reflect how nodes of different degrees impact the routing process in the complicated VANET scenario.

IV. GCN-based Greedy Routing

Our proposed SDVN routing algorithm is divided into two parts: the centralised and the distributed parts. Each part is deployed on the two components of the hybrid SDVN (i.e., the control plane and the data plane). The control plane, which includes the central controller, BSs, and RSUs, is responsible for the centralised part. While the data plane, which consists of vehicles and OBUs mounted on them, is responsible for the distributed part. Based on the powerful learning capacity of NiGCN stated in Section III and the global vision brought by SDVN, we propose NGGRA.

A. Model setting and data collection

Since NGGRA uses the NiGCN model, an appropriate feature-label setting is necessary for the training. At first, as the basis of the model training, the training data collection will be the first activity. Every vehicle needs to record and store the following information to establish the features and labels.

Position $P(P_x, P_y)$: The position of the source node and the destination node will be recorded. We use the source node's positions as the features and record the destination node's positions for the corresponding model training.

Velocity $V(V_x, V_y)$: The velocity of the source vehicle will be stored and utilised as a feature, both the speed on the x-axis and the y-axis. These two features represent the driving direction of the vehicle.

Acceleration $A(A_x, A_y)$: We compute the acceleration on each axis by comparisons with the velocity of the source vehicle itself at the last moment.

Vehicle load L: All vehicles are assumed to have a fixed (and limited) capacity (spectrum efficiency affected by transmission power, distance, and rate threshold). Once we can evenly distribute routing tasks on each vehicle, the overall quality of data transmission will be higher. In inverse, once the load of several vehicles is high or even full, the data communications for these vehicles will be difficult to continue. This feature is represented as the capacity to handle the transmission task.

Delay D: All vehicles record the end-to-end delay of the routing process. This metric can directly reflect the quality of this routing path. It is also the objective of our routing. If this routing transmission fails, we record it as infinity. Thus, we record and utilise this information as a factor of the labels.

For every successful or failed routing path, the related vehicle v_i will send the aforementioned routing-related information to the controller as training data via RSUs and BSs. Positions, velocity, acceleration, and vehicle load will be utilized as features $f_i(P_i, V_i, A_i, L_i)$. Since NiGCN can solve multi-classification issues, we need to process the delay to make it a classification issue. The process is presented as follows:

As a prerequisite, we will divide the coverage area into several grids $(g_0, g_1, \dots g_n)$ and train a NiGCN model for each grid according to the destination node's position. We will define the label of each NiGCN model as follows:

DEFINITION 1 (transmission rank (τ_i^d)**):** *We define* τ_i^d *to represent the level of difficulty for the current vehicle* v_i *to transmit packets to the destination vehicle which is located in grid* g_d*. This rank will be divided into four levels starting from 1 to 4. The larger levels represent the growing difficulty in successfully transmitting the packet to the destination.*

As we can see, τ_i^d corresponds exactly to $\tau_{v_i}(v_d, t)$ in Eq. (11) mentioned in Section II. The delay information collected from the data plane is the key to computing the label for the training. For all routing information, we define a value called delay performance: $\delta = d/dis_{src,des}$. d is the delay of this routing, and $dis_{src,des}$ is the distance between the source and destination vehicle in this corresponding routing, which we can easily obtain from the location information. Then, according to the grid where the destination vehicle is located, different routing information will be classified to train the corresponding NiGCN model. Then, among all the routing records for each grid g_d, the first 10% of the delay performance δ is τ_i^d 1, the $10\% - 40\%$ of the delay performance is τ_i^d 2, and 40%-100% of the delay performance is τ_i^d 3. The failed routing paths will be recorded as τ_i^d 4. τ_i^d represents the difficulty for the source vehicle v_i in transmitting data to a specific grid g_d. Furthermore, it will be the labels of the output of the NiGCN model. The number of τ levels is important. If this number is small, it cannot distinguish the relay qualifications of neighbours fine-grained. If the number is big, it is hard for the NiGCN to converge. After the simulations, 4 is the best number here.

Each source and relay vehicle will update its features, packet destination, and timestamp to the control plane. Once the corresponding destination vehicle receives the packet, it will upload its information too. The detail of data collection is introduced in Algorithm 1. The control plane can efficiently compute the delivery delay with this information. Except for these, the controller will classify the routing information whose occurrence time is closer and construct an adjacency matrix according to their positions, which serves as the graph-related training data in NiGCN.

Thus far, we have completed preparing the training data for NiGCN. However, we still have another training model, a multi-layer feedback neural network for the relay vehicle selection. The features are almost the same, and the position, velocity, acceleration, and load will be recorded as features. Meanwhile, we need to collect these features of the next-hop node (f_n) of the source node (f_s) on the routing path. In addition, according to the destination grid g_d of this routing path, we add the τ_n^d of this next-hop node v_n obtained from NiGCN into features. It is worth noting that we do not set the destination node information as features here. We can know everything we want from τ_n^d achieved through NiGCN. We define the labels for this model as follows.

DEFINITION 2 (relay rank $(\sigma_i^{s,d})$**):** *We define* $\sigma_i^{s,d}$ *as the suitability for the vehicle* v_i *to transmit packets as the relay node from the source vehicle* v_s *to the destination vehicle which is located in grid* g_d*. This rank will be divided into four levels starting from 1 to 4. The smaller levels represent that it is more appropriate to use* v_i *as a relay to transmit data from* v_s *to* g_d*.*

$\sigma_i^{s,d}$ corresponds to $\sigma_{v_i}(v_j, t)$ in Eq. (12). Labels will also be constructed based on the delay performance δ. The first 10% of the delay performance δ is $\sigma_i^{s,d}$ 1, the $10\% - 40\%$ of the delay performance is $\sigma_i^{s,d}$ 2, and 40%-100% of the delay performance is $\sigma_i^{s,d}$ 3. The failed relay neighbours will be recorded as $\sigma_i^{s,d}$ 4. We use the traditional feedback neural network model to achieve this rank. For the training data collection of this model, we rank all the neighbours with the qualifications of the relay vehicles within the same source and destination vehicles. Thus, we randomly select some source nodes during the data transmission. These sources will broadcast the packet to all its neighbours and upload the IDs of these neighbours and the aforementioned uploaded information. Additionally, these neighbours will transmit the duplicate packets to the destination vehicle. This method will have multiple routing paths from the source to the destination. Once the destination obtains these duplicate packets, the destination vehicle will upload all the routing-related information of packets relayed by different neighbours to the controller for the multi-layer feedback neural network training. We present the input and output of these two models in Fig. 3. The detail of data collection is introduced in Algorithm 1.

After the daily data collection, once the training data size is adequate, the controller will train the NiGCN models and a multi-layer feedback neural network model and then distribute them to each vehicle through BSs and RSUs. Before the training data is collected, we will use the old model in a routing computation. We will use GPSR as our routing protocol during the initialisation period when no model has been trained.

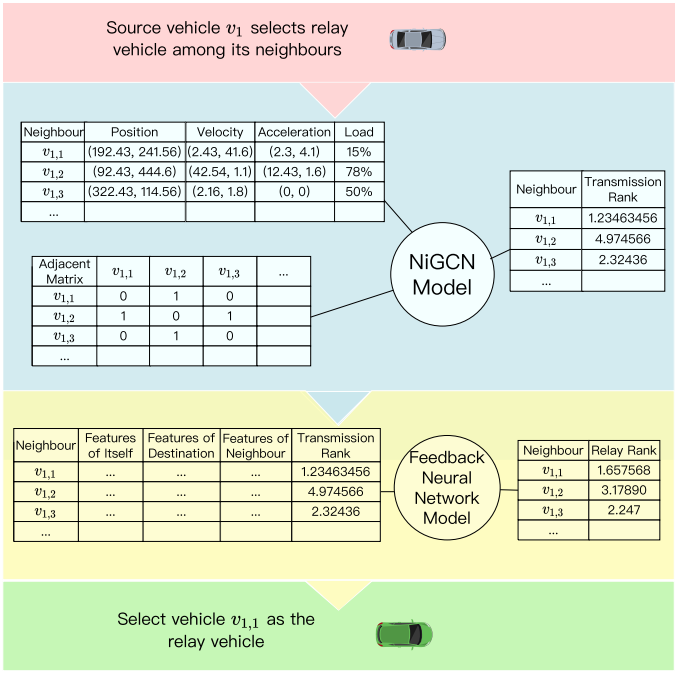

Fig. 3. The detailed inputs and outputs of two models.

B. NGGRA process

As a prerequisite, all vehicles will periodically exchange their features with their neighbours during the routing calculation. When one vehicle v_i receives the features from its neighbours, it needs to use the first layer of NiGCN to achieve H_i^1 through (19). Then v_i will attach H_i^1 to the ACK message and broadcast it. In this method, each vehicle has its neighbours' H_n^1. It is vital for NiGCN since this model needs two-hop neighbour information. If vehicles regularly maintain their one-hop neighbour information, they can achieve two-hop neighbour information efficiently with the one-hop broadcast once they have transmission requests.

As we can see in Algorithm 1, once the vehicle needs to transmit data, it uploads routing-related information to the control plane at first, including features, destination, and time stamp (Line 3). Then the source vehicle will be selected as the broadcasting vehicle to collect data for the neural network(NN) training under a certain probability. We set this probability as 5%. After tests, it can collect adequate training data while avoiding increasing overhead. It broadcasts packets to its neighbours and uploads neighbours' information (Lines 5-6). If the source vehicle has not been selected, it computes the grid number where the destination vehicle is located g_d (Line 9). Then, it broadcasts a routing request to all its neighbours (Line 10). This request contains its features f_s and the destination grid g_d. Then, suppose that the vehicle v_i receives the routing request as a neighbour (Line 13). In that case, it achieves the first layer output of itself and its neighbours, $H_{i,d}^1$ and $H_{n,d}^1$, with different NiGCN models according to g_d. Then v_i utilises the corresponding (20) in the NiGCN models they have stored to obtain their transmission ranks τ_i^d (Line 18). Then, τ_i^d is concatenated with their features f_i, and the features of the source vehicle f_s as the input to the multi-layer feedback neural network. After the computation, the relay rank $\sigma_i^{s,d}$ can be obtained (Line 19), and the neighbour v_i sends this $\sigma_i^{s,d}$ back to the source v_s (Line 20). The source selects the optimal one as the next-hop node and sends the packets (Lines 23-26).

By repeating this process greedily (Line 36), vehicles transmit the packets towards the destination through the vehicular network. Once the vehicle holding packets finds the destination vehicle in the neighbouring information of the vehicles with the packets, it transmits the packets using the knowledge it contains. The routing process is completed (Lines 14-16 and 28-31). And the destination node will upload routing-related information (Line 29). If the vehicle holding the packets is in the same grid as the destination vehicle, GPSR will forward the packet (Line 33). With a reasonable grid division strategy according to the transmission range, packets can be easily transmitted to the destination within two hops under GPSR.

As stated in Section II(B), our goal is to maximise the delivery ratio and minimise transmission delay. Two decision models are fully designed to satisfy these two goals. The NiGCN model aims to achieve transmission rank τ_i^d of the neighbour vehicle v_i. In this model, the labelling process entirely depends on the delay performance. The packet loss situation will be viewed as the infinity long delay and labelled as the worst rank during the data collection stage. Under this labelling system, we can evaluate the delivery ratio and delay simultaneously, and the total transmission quality relies entirely on these two metrics according to Section II(B). Thus, we can view this model as a function to achieve the transmission quality from vehicle v_i to destination d based on these two vehicles' features. However for the other model $NN(f_s, f_i, \tau_i^d)$, the qualification for v_i to transmit data to the destination as a relay node can be well assessed as relay rank $\sigma_i^{s,d}$. Since we select transmission rank as one of the features, the model can learn from the transmission quality from v_i to d. Meanwhile, this model considers the transmission quality from the actual source vehicle v_s and its neighbour v_i. Thus, this model considers the transmission quality of the link between the source node and the relay node and evaluates the capacity of this relay node to transmit data packets to the destination. Similarly, the packet loss situation is also considered during the label construction stage. Since all training data of these two models are fully collected from the data plane, the model's training will consider the global view. These decision models are suitable for all vehicles in the covered area. Based on these two decision models, $NiGCN_d(f_i)$ and $NN(f_s, f_i, \tau_i^d)$, each vehicle is able to select the optimal relay vehicle among its neighbours, considering the optimal delivery ratio and delay simultaneously.

V. Simulation

The simulation is divided into two parts: the NiGCN and the routing. The first part focuses on node classification tasks for transductive learning based on the standard dataset. The simulation results demonstrate that the NiGCN exhibits

Algorithm 1: $NGGRA(v_i, v_s, v_d)$

Input: $f_i(P_i, V_i, A_i, L_i)$: the feature of vehicle v_i; $H^1_{i,d}$: the first layer output of v_i with d NiGCN model; $NiGCN_d(f_i)$: the trained NiGCN model for g_d; $NN(f_s, f_i, \tau^d_i)$: the trained multi-layer feedback neural network model;

switch v_i **do**
  case v_i *needs to transmit* $packet(v_i, v_d, data)$ *to* v_d **do**
    Upload routing-related information $\rho(packet, f_i, time)$ to the control plane;
    if $random() \leq 0.05$ **then**
      Upload stored neighbourhood information $(v_{i,1}, v_{i,2}, \ldots . v_{i,n})$;
      Broadcast $packet(v_i, v_d, data)$;
    end
    else
      Calculate the grid g_d which v_d belongs to;
      Broadcast the routing request $req(v_i, f_i, g_d, v_d)$;
    end
  end
  case v_i *receive the routing request* $req(v_s, f_s, g_d, v_d)$ **do**
    if $v_i == v_d$ **then**
      Transmit $\sigma^{s,d}_i = 0$ to v_s;
    end
    else
      $\tau^d_i = NiGCN_d(H^1_{i,d}, H^1_{n,d})$;
      $\sigma^{s,d}_i = NN(f_s, f_i, \tau^d_i)$;
      Transmit $\sigma^{s,d}_i$ to v_s;
    end
  end
  case v_i *receive relay rank collection* $\zeta\{\sigma^{i,d}_1, \sigma^{i,d}_1, \ldots, \sigma^{i,d}_n\}$ *from its neighbourhood* $(v_{i,1}, v_{i,2}, \ldots . v_{i,n})$ **do**
    Select the optimal one $\sigma^{i,d}_o$ from ζ;
    Transmit $packet(v_s, v_d, data)$ to v_s;
  end
  case v_i *receives* $packet(v_s, v_d, data)$ **do**
    if $v_i == v_d$ **then**
      Upload routing-related information $\rho(packet, f_i, time)$ to the control plane;
      return;
    end
    else if $g_i == g_d$ **then**
      $GPSR(v_i, v_s, v_d)$;
    end
    else
      $NGGRA(v_i, v_s, v_d)$;
    end
  end
end

improvements or performs similar to the popular studies, and the training efficiency is better than the other methods. In the other part, we run the data transmission simulation of the vehicular network in our open-source simulator based on a real-world scenario (https://github.com/a824899245/SDVN-platform). We compute and compare several routing-related metrics with several greedy-based prior studies, including the delivery delay, delivery ratio, average hop count, and throughput.

A. NiGCN Simulation

This section evaluates our proposed NiGCN on node classification tasks with a transductive learning setting. Under both transductive learning settings, the features of the unlabeled data are accessible and available during training. Specifically, for node classification, only a part of the nodes in the graph are labelled. However, other nodes in the same graph are accessible during training, including their features and links, except for the labels. Thus, the training process knows about the graph structure that contains the testing nodes. We use three standard benchmark datasets for transductive learning experiments from Cora, Citeseer, and Pubmed [24]. These three datasets are citation networks with nodes and edges representing documents and citations. Each node's feature vector corresponds to a document's bag-of-word representation. For these three datasets, we employ the same experimental settings as those in GCN [23]. The Cora dataset contains 2708 nodes, 5429 edges, 7 classes, and 1433 features per node. The Citeseer dataset contains 3327 nodes, 4732 edges, 6 classes, and 3703 features per node. The Pubmed dataset contains 19717 nodes, 44338 edges, 3 classes, and 500 features per node. The specific $train\backslash validation\backslash test$ split can be found in [25]. All data settings are subject to the downloaded data without any modifications.

1) Experimental Setup: In the transductive learning tasks, we employ the proposed NiGCN models as illustrated in Fig. 2. Since the transductive learning datasets employ high-dimensional bag-of-words representations as feature vectors of nodes, the inputs go through a graph embedding layer to reduce the dimension. We use a GCN layer with a node importance strategy as the graph embedding layer as Eq. (18). We also utilise the multi-node-importance strategy here. K is set to be 3. Finally, these three embeddings will be concatenated, and they are the input of a traditional GCN layer as Eq. (20). Dropout [26] is applied to both the input feature vectors and the adjacency matrices in each layer with rates of 0.16 and 0.999, respectively. The sub-graph training strategy in [25] is used on the Pubmed. The sub-graph size is 2000.

We use the identity activation function for all layers, which means no nonlinearity is involved in the networks. In order to avoid over-fitting, the L2 regularisation with $\lambda = 0.0005$ is applied. For training, we use the Adam optimiser [27] with a learning rate of 0.05. The Glorot initialisation initialises the weights in NiGCNs [28]. We employ an early stopping strategy based on the validation accuracy and train 1,000 epochs.

2) Analysis of Results: Table II summarises the results of our comparative evaluation experiments. Our results successfully demonstrate the best performance being achieved or matched across all three datasets according to our expectations. Our NiGCN models achieve better performance over the current popular GCNs by a margin of 0.6% when using the Cora datasets. In addition, at the same time, on the other datasets, NiGCN can achieve almost the same performance as other studies. The model can learn the node importance through the simple concatenation between degree and feature matrix, and the accuracy can be improved with a slight increase in the training complexity. Moreover, different from the concept of a fixed neighbourhood in [25], [29], we apply no prejudice before training. NiGCN learns all information learned from the full feature and entire graph structure.

To evaluate our advantage in the training efficiency, we ran the three datasets on all four models. The epoch number is 1000. No batch strategy is utilised in this part of the simulation. We can see that our model has much better training efficiency than the other models due to the simplicity of our model. Our model with a simple structure achieves good performance efficiently. That means our model can adapt to the rapidly evolving vehicular network topology. Meanwhile, GAT concatenates the node's and its neighbour's features and trains on them to achieve knowledge of the link [18]. GAT can smoothly achieve node importance during the training. However, there could be additional trainable weights for the attention coefficients training, decreasing training efficiency.

B. Routing Simulation

In this subsection, we present the details of the routing simulation, including the parameters and evaluations. We run the simulation on our open-source simulator to meet the simulation demands of SDVNs (https://github.com/a824899245/SDVN-platform), i.e., a powerful tool developed in the Python environment. Our simulation is based on real maps of the Tiexi District, Shenyang City, China. The sizes of the two maps are $2686m \times 1494m$ and $5193m \times 5863m$, respectively (specific satellite map images can be obtained in the GITHUB project). The map information was obtained from the OpenStreetMap [30]. Then, we apply SUMO [31] to generate the movement data of the vehicle to fit on these maps. Last, we use our simulator to perform the data packet transmission in the vehicular networks. 20% of the vehicles will randomly generate routing requests once per second. Besides, the destination of the request is random. We set the average distance between the random source and the destination vehicle around half of the diagonal length of the map. Table III shows the parameter settings of our vehicular network and simulator. Each round of the simulation lasts for 300 seconds. Each simulation round is repeated with different random seeds five times, while four routing algorithms use the same seeds at each round.

We compare our routing algorithm, NGGRA, with HRLB [9], QGRID [10], and SeScR [32]. The first simulation's counterpart is an SDVN greedy routing scheme HRLB [9]. HRLB is a hierarchical routing scheme that considers load balancing and adopts a three-level architecture to compute the routing path. First, at the grid selection stage, HRLB considers the average grid vehicle density and the average grid transfer probability. Then, the segments are connected based on the selected grids. Finally, the qualified vehicles on the related segments will be selected as a part of the optimal routing path. HRLB also adopts the SDVN architecture and uses the greedy strategy in each layer to select the next-hop elements. We conclude that it is a greedy-based routing algorithm under the SDVN architecture, similar to our algorithm. Hence, we select this as one of our counterparts.

QGrid is a reinforcement learning-based hierarchical protocol [10]. The protocol is also hierarchical. Firstly, it divides the geographic area into smaller grids and finds the next optimal grid towards the destination by Q-learning based on the vehicle density. Secondly, it discovers a vehicle inside or moving towards the next optimal grid for message relaying. During the vehicle selection, QGrid also uses the greedy strategy and uses the machine learning method. So this routing algorithm could be a good counterpart. We select $QGrid\_G$ without bus-aided since there is no specific vehicle type setting in the data, such as taxi or bus. The data update time slot $\triangle T$ is the 20s, and the lifetime of each message TTL is 50.

The last counterpart is SeScR. It is an SDN-enabled spectral clustering-based routing using deep learning [32]. At first, the vehicles are categorised into clusters using the eigenvalues of graph laplacian. Each cluster head is elected by affinity matrix and distance model. The second phase uses SDN and the deep deterministic policy gradient (DDPG) algorithm for routing computation. An actor-critic architecture for the continuous address space is utilised to find a set of optimal routing paths. Since this algorithm uses the SDVN architecture and deep learning methods, we select SeScR as our counterpart.

To compare the above routing schemes, we choose four evaluation metrics with considerations of various requirements for routing quality [33], which are packet delivery ratio, delivery delay, average hop count, and throughput.

Packet delivery ratio: The ratio of data packets successfully delivered to destination nodes among all generated data packets.

Delivery delay: The average end-to-end delay from when a packet is created to when it is delivered to the destination. This metric primarily shows the quality of the computed routing path.

Average hop count: The average number of hops the packets transmitted from the source node to the destination node. It is a vital metric of the routing algorithm's capacity for analysing the vehicular network's status. It has a significant impact on delivery delays.

Throughput: The average number of packets successfully transferred from source nodes to destination in unit time in the network during the whole simulation.

As we can see in Fig. 4(a) and Fig. 5(a), NGGRA can achieve excellent performance on the delivery ratio in most

TABLE II
ACCURACY AND TRAINING EFFICIENCY RESULTS OF LEARNING EXPERIMENTS IN TERMS OF NODE CLASSIFICATION ACCURACIES ON THE CORA, CITESEER, AND PUBMED DATASETS.

Models	Cora	Citeseer	Pubmed
GCN [23]	$81.5/\mathbf{70}s$	$70.5/\mathbf{85}s$	$79/\mathbf{380}s$
GAT [18]	$83.0 \pm 0.7/158s$	$72.5 \pm 0.7/650s$	$79.0 \pm 0.3/1080s$
LGCN [25]	$83.3 \pm 0.5/574s$	$\mathbf{73.0 \pm 0.6}/668s$	$\mathbf{79.5 \pm 0.2}/2542s$
NiGCN	$\mathbf{83.9 \pm 0.7}/90s$	$72.6 \pm 0.7/104s$	$79.2 \pm 0.7/470s$

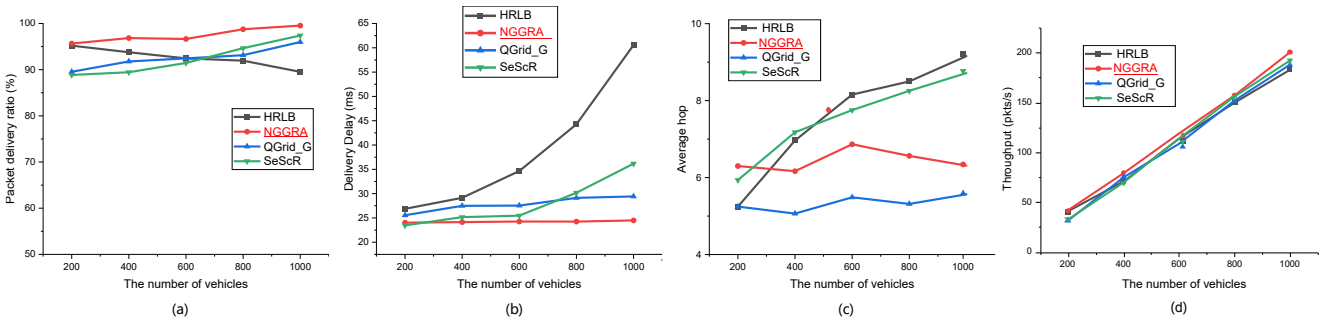

Fig. 4. Four metrics versus the number of vehicles on the small small-area map. (a) Packet delivery ratio $(\%)$, (b) Packet delivery delay (ms), (c) Average hop count, (d) Throughput $packets/s$

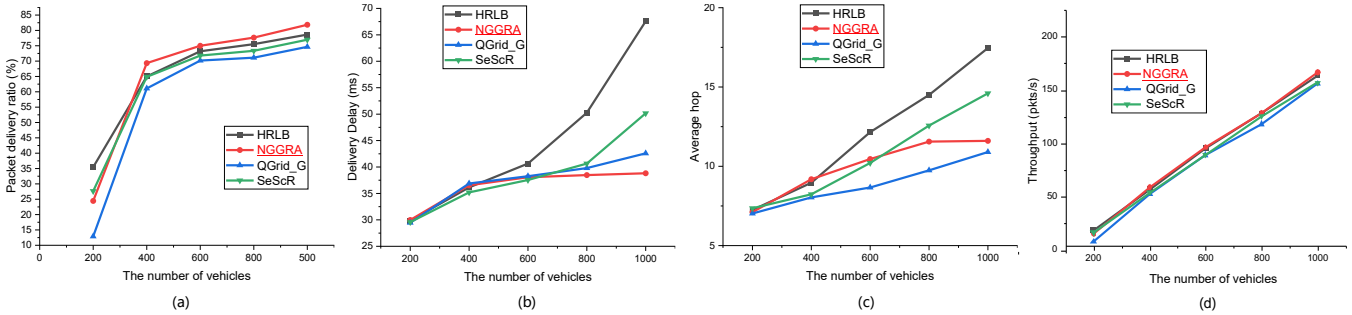

Fig. 5. Four metrics versus the number of vehicles on the large-area map. (a) Packet delivery ratio $(\%)$, (b) Packet delivery delay (ms), (c) Average hop count, (d) Throughput $packets/s$

scenarios through a well-trained NiGCN model with full consideration of various network-related parameters. The advantage of our method in terms of the delivery ratio can be comprehensive compared with $QGrid\_G$ and SeScR. For $QGrid\_G$, this is an inevitable result since QGrid only considers the vehicle density during the deep learning training process. Meanwhile, a simple greedy strategy is utilised during the vehicle selection part in QGrid. The connection between grid path and vehicle traffic is complex. Therefore, in the case of low vehicle density (200 vehicles on the small map, 200 vehicles on the large map), it is difficult for QGrid to find a feasible routing path under the calculated grid path. An unusual phenomenon is the low packet delivery ratios of all four routing algorithms in the low-density scenario in the large area scenario. The reason is that two hundred vehicles are too sparse for the area of $5193m \times 5863m$. In this case, it is hard to find a relay vehicle to forward data messages in such a sparse scenario, which leads to packet loss. Thus, all algorithms do not perform well in an overly sparse network. HRLB and SeScR are two centralised routing algorithms under the traditional SDVN architecture, which perform better in this sparse scenario. These better performances are due to the global view of the SDVN. All vehicles must upload their statuses to the controller under the traditional SDVN routing architecture. The controller can compute the globally optimal routing path for the vehicle. More forwarding opportunities are available in the low-density vehicle network under this scheme. However, as the density of vehicles increases, the capacity for analysing the network deteriorates, and the performance of the delivery ratio gradually decreases in HRLB. It is because there is no right way to avoid local optima in HRLB. Meanwhile, the traditional SDVN routing architecture leads to an increase in the delay due to the information exchange between vehicles and controllers during the routing computation, as shown in Fig. 4(b) and Fig. 4(b). The powerful learning capacity of

TABLE III
SIMULATION PARAMETERS.

Simulation Parameter Name	Value
Size of the simulation area	$2686m \times 1494m$ $5193m \times 5863m$
Intersections	$68/267$
Road segments	$116/457$
Number of vehicles	$200/400/600/800/1000$
Vehicle velocity	$0 - 60km/h$
Vehicle transmission range	$500m$
Grid Diagonal Length	$1000m$
Simulation duration	$300s$
Data Packet Size	1024 Bytes
Standards	IEEE 802.11p
Buffer Size	20 packets

NiGCN and the global information brought by the SDVN architecture enables NGGRA to achieve good performance in various scenarios.

The advantages of NGGRA can be more clearly seen in Fig. 4(b)&(c) and Fig. 5(b)&(c). Since $QGrid\_G$ routing path computation depends on the optimal grid sequence based on the Q-learning. It is worth indicating that the length of the grid sequence is limited. This mechanism makes the average hop count of the computed routing path low and stable. Thus, on the metric of the average hop, $QGrid\_G$ has the best performance in most scenarios. However, for the delivery delay, the disadvantages caused by $QGrid\_G$ gradually appear on the aspect of the metric that truly impacts the main factor of QoS. Since this algorithm only considers vehicle density during the grid sequence generation process, it is only viewed as a weight parameter. Meanwhile, they use the traditional greedy strategy and only consider distance during the following vehicle selection. Thus, this algorithm may not accurately evaluate the high-dynamic vehicular network. It is the main reason for the disadvantage of the delay for the QGrid.

Fig. 4(d) and Fig. 5(d) reflects the throughput of the four protocols. The throughputs of four protocols all increase with the number of generated packets and delivery ratio. Moreover, NGGRA achieves the best performance. It can achieve an average gain of about 8.5, 6.9 and 7.8 packets per second in the small map and 8.6, 4.8, and 0.5 packets per second in the large map over QGriD, SeSCR, and HRLB. The main reason is that our protocol can deliver packets with a high success rate. Also, semi-centralised network architecture greatly reduces the network load so that it can still complete data transmission stably in the case of rapid network traffic growth.

For our algorithm NGGRA, in the low-density scenario, the source vehicle needs to achieve the relay rank from the neighbour. This part of the increment on the delivery delay makes the advantage of NGGRA less evident with the sparse scenario. However, with the increase in the number of vehicles, NGGRA achieves the best and most stable performance in terms of delivery delay and has only a limited increase in the average hop. It is the result of two reasons: First, we comprehensively consider the various network-related parameters of the vehicles. We take the velocity and acceleration into the model's construction. These two parameters help us better predict the vehicle's position after the training. We also consider the vehicle load, a parameter that significantly impacts the delay. NGGRA achieves a better evaluation of the vehicular network based on these parameters.

Conversely, QGrid only considers the distance and vehicle density. This limited consideration can bring a minimal and stable hop count. However, the ignorance of other vital network-related parameters makes it unable to evaluate the network sufficiently and inevitably decreases the QoS. On the other hand, a well-designed and trained NiGCN model improves the capacity for analysing these parameters. The vehicle's qualifications for transmitting these packets to the destination can be sufficiently evaluated through this model, significantly contributing to selecting the optimal relay vehicle for each vehicle. The computed routing path may cause a small increase in the average hop. However, our computed routing path can certainly guarantee a good and stable QoS due to the various parameters and the assistance of NiGCN.

However, due to the traditional SDVN routing architecture, the routing requests and replies between the vehicle and controller will increase the delay. Since each vehicle must periodically transmit its statuses to the controller, this mechanism will burden the network. This increase in the overhead will inevitably lead to an increase in the delivery delay and network load. Thus, when the vehicle density increases and the network becomes complicated, HRLB and SeScR cannot achieve good and stable performance in terms of delay. Especially for HRLB, all three layer selections used the greedy strategy and adopted no mechanism to avoid local optimality. As for the SeScR, clustering efficiency becomes lower, and convergence becomes slower with the increase in the vehicle density. A complicated clustering situation inevitably leads to increased delay and unsuccessful routing. Thus, for the traditional SDVN routing, an algorithm that can efficiently and accurately solve routing requests in any complex situation is necessary.

C. Summary of Performance Results

In conclusion, we designed a series of simulations to evaluate the performance of the NiGCN model and routing algorithm NGGRA. Compared with three popular studies, it has been indicated that the NiGCN is a powerful model for graph learning. It has achieved excellent performance in each dataset, and the training speed surpasses other works. Due to its simplicity and strong learning capacity, this model is a powerful tool for learning graph-related structural features.

In the second group of experiments, NGGRA remains high in all the different scenarios in terms of all four metrics for

evaluating the routing quality. NGGRA is the optimal routing algorithm for delivery ratio, delay, and throughput. In other cases, it is at least similar to the optimal algorithm. The simulation experiments confirm that NGGRA can efficiently compute the routing path with high quality. These simulations conclude that NGGRA is qualified enough to handle redundant routing requests in the increasingly complex vehicular network.

VI. Conclusion

This paper proposes a novel graph convolution network model, NiGCN, and a new robust routing algorithm NGGRA based on this model. We consider a vital factor, node importance, during the processing of constructing the NiGCN model. Besides, the degree of each node represents the node importance. These features have also been naturally integrated into the model. This novel routing algorithm, NGGRA, is based on the hybrid SDVN architecture and centralised NiGCN model. In NGGRA, the source vehicle obtains the transmission and relay ranks of all its neighbours based on the NiGCN model. Then, this vehicle selects the optimal relay vehicle based on the features of both sides and the transmission rank. The features include positions, velocities, acceleration, and loads. Vehicles will repeat this selection process iteratively and greedily, and finally, the packet reaches the destination. We have performed extensive simulation experiments to evaluate the performance of NiGCN in accuracy and training speed. The performance is better than the other existing models. In the routing part, we compare NGGRA with several other schemes in terms of delivery ratio, delivery delay, average hop, and throughput. Built upon the powerful NiGCN model and appropriate routing algorithm setting, NGGRA outperforms other counterparts in terms of the delivery ratio, throughput, and delay. NGGRA sufficiently guarantees the QoS of the obtained routing path under different scenarios.

Acknowledgment

This paper is supported in part by the funds of the Department of Science and Technology of Liaoning Province, in part by the Science Foundation of Liaoning Province under Grant 2020-MS-237, and in part by the Digit Fujian Internet-of-Things Laboratory of Environmental Monitoring Research Fund (Fujian Normal University) under Grant 202001.

References

[1] G. Karagiannis, O. Altintas, E. Ekici, G. Heijenk, B. Jarupan, K. Lin, and T. Weil, "Vehicular networking: A survey and tutorial on requirements, architectures, challenges, standards and solutions," *IEEE communications surveys & tutorials*, vol. 13, no. 4, pp. 584–616, 2011.

[2] A. Hawbani, E. Torbosh, W. Xingfu, P. Sincak, L. Zhao, and A. Y. Al-Dubai, "Fuzzy based distributed protocol for vehicle to vehicle communication," *IEEE Transactions on Fuzzy Systems*, 2019.

[3] G. Karagiannis, O. Altintas, E. Ekici, G. Heijenk, B. Jarupan, K. Lin, and T. Weil, "Vehicular networking: A survey and tutorial on requirements, architectures, challenges, standards and solutions," *IEEE Communications Surveys Tutorials*, vol. 13, no. 4, pp. 584–616, 2011.

[4] B. T. Sharef, R. A. Alsaqour, and M. Ismail, "Vehicular communication ad hoc routing protocols: A survey," *Journal of network and computer applications*, vol. 40, pp. 363–396, 2014.

[5] B. Karp and H.-T. Kung, "GPSR: Greedy perimeter stateless routing for wireless networks," in *Proceedings of the 6th annual international conference on Mobile computing and networking*, 2000, pp. 243–254.

[6] X. Hu, L. Ma, Y. Ding, J. Xu, Y. Li, and S. Ma, "Fuzzy logic-based geographic routing protocol for dynamic wireless sensor networks," *Sensors*, vol. 19, no. 1, 2019. [Online]. Available: https://www.mdpi.com/1424-8220/19/1/196

[7] R. Attia, A. Hassaan, and R. Rizk, "Advanced greedy hybrid bio-inspired routing protocol to improve iov," *IEEE Access*, vol. 9, pp. 131 260–131 272, 2021.

[8] T. Nebbou and M. Lehsaini, "Greedy curvemetric-based routing protocol for vanets," in *2018 International Conference on Selected Topics in Mobile and Wireless Networking (MoWNeT)*, 2018, pp. 1–6.

[9] Y. Gao, Z. Zhang, D. Zhao, Y. Zhang, and T. Luo, "A hierarchical routing scheme with load balancing in software defined vehicular ad hoc networks," *IEEE Access*, vol. 6, pp. 73 774–73 785, 2018.

[10] F. Li, X. Song, H. Chen, X. Li, and Y. Wang, "Hierarchical routing for vehicular ad hoc networks via reinforcement learning," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 2, pp. 1852–1865, 2018.

[11] Z. Zhang, P. Cui, and W. Zhu, "Deep learning on graphs: A survey," *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[12] D. Kreutz, F. M. V. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, 2014.

[13] L. Zhao, K. Yang, Z. Tan, X. Li, S. Sharma, and Z. Liu, "A novel cost optimization strategy for sdn-enabled uav-assisted vehicular computation offloading," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 6, pp. 3664–3674, 2020.

[14] L. Zhao, A. Al-Dubai, A. Y. Zomaya, G. Min, A. Hawbani, and J. Li, "Routing schemes in software-defined vehicular networks: Design open issues and challenges," *IEEE Intell. Transp. Syst. Mag.*, 2020.

[15] M. M. Islam, M. T. R. Khan, M. M. Saad, and D. Kim, "Software-defined vehicular network (SDVN): A survey on architecture and routing," *Journal of Systems Architecture*, vol. 114, p. 101961, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1383762120302113

[16] M. Jerbi, R. Meraihi, S.-M. Senouci, and Y. Ghamri-Doudane, "Gytar: improved greedy traffic aware routing protocol for vehicular ad hoc networks in city environments," in *Proceedings of the 3rd international workshop on Vehicular ad hoc networks*, 2006, pp. 88–89.

[17] A. Akhunzada and M. K. Khan, "Toward secure software defined vehicular networks: Taxonomy, requirements, and open issues," *IEEE Communications Magazine*, vol. 55, no. 7, pp. 110–118, 2017.

[18] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.

[19] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[20] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[21] L. Tan, K. Yu, A. K. Bashir, X. Cheng, F. Ming, L. Zhao, and X. Zhou, "Toward real-time and efficient cardiovascular monitoring for covid-19 patients by 5g-enabled wearable medical devices: a deep learning approach," *Neural Computing and Applications*, pp. 1–14, 2021.

[22] X. Li, S. Han, L. Zhao, C. Gong, and X. Liu, "New dandelion algorithm optimizes extreme learning machine for biomedical classification problems," *Computational intelligence and neuroscience*, vol. 2017, 2017.

[23] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[24] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad, "Collective classification in network data," *AI magazine*, vol. 29, no. 3, p. 93, 2008.

[25] H. Gao, Z. Wang, and S. Ji, "Large-scale learnable graph convolutional networks," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 1416–1424.

[26] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[27] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[28] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 2010, pp. 249–256.
[29] M. Zhang, Z. Cui, M. Neumann, and Y. Chen, "An end-to-end deep learning architecture for graph classification," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.
[30] M. Haklay and P. Weber, "Openstreetmap: User-generated street maps," *IEEE Pervasive Computing*, vol. 7, no. 4, pp. 12–18, 2008.
[31] D. Krajzewicz, J. Erdmann, M. Behrisch, and L. Bieker, "Recent development and applications of SUMO-Simulation of Urban MObility," *International journal on advances in systems and measurements*, vol. 5, no. 3&4, 2012.
[32] A. Nahar and D. Das, "Sescr: Sdn-enabled spectral clustering-based optimized routing using deep learning in vanet environment," in *2020 IEEE 19th International Symposium on Network Computing and Applications (NCA)*. IEEE, 2020, pp. 1–9.
[33] L. Zhao and A. Y. Al-Dubai, "Routing metrics for wireless mesh networks: a survey," in *Recent Advances in Computer Science and Information Engineering*. Springer, 2012, pp. 311–316.

Zhuhui Li received the M.S. degree in Computer Science and Technology from Shenyang Aerospace University, China. His research interests mainly include VANETs, FANETs, SDVN, GCN, and Temporal Graph.

Liang Zhao [M] is a Professor at Shenyang Aerospace University, China. He received his Ph.D. degree from the School of Computing at Edinburgh Napier University in 2011. Before joining Shenyang Aerospace University, he worked as associate senior researcher in Hitachi (China) R&D from 2012 to 2014. His research interests include ITS, VANET, WMN and SDN. He has published more than 100 peer-reviewed papers. He served as the Chair and Co-Chair of more than 20 international conferences and workshops such as 2021 IEEE TrustCom (Program Co-Chair) and 2020 IEEE IUCC (General Co-Chair). He is/has been a guest editor of journals such as IEEE Transactions on Network Science and Engineering, Springer Journal of Computing. He was the recipient of the Best/Outstanding Paper Awards at IEEE IUCC 2015, ACM MOMM 2013, and IEEE ISPA 2020.

Geyong Min is a professor of high-performance computing and networking in the Department of Mathematics and Computer Science within the College of Engineering, Mathematics and Physical Sciences at the University of Exeter, United Kingdom. He received his Ph.D. degree in computing science from the University of Glasgow, United Kingdom, in 2003, and his B.Sc. degree in computer science from Huazhong University of Science and Technology, China, in 1995. His research interests include future Internet, computer networks, wireless communications, multimedia systems, information security, high-performance computing, ubiquitous computing, modeling, and performance engineering. His postal address is University of Exeter, Exeter, United Kingdom.

Ahmed Y. Al-Dubai [SM] is Professor of Networking and Communication Algorithms in the School of Computing at Edinburgh Napier University, UK. He received the PhD degree in Computing from the University of Glasgow in 2004. His research interests include Communication Algorithms, Mobile Communication, Internet of Things, and Future Internet. He received several international awards. His postal address is 10 Colinton Road, Edinburgh, United Kingdom.

Ammar Hawbani [M] received the B.S., M.S. and Ph.D. degrees in Computer Software and Theory from the University of Science and Technology of China (USTC), Hefei, China, in 2009, 2012 and 2016, respectively. Currently, he is a postdoctoral researcher in the School of Computer Science and Technology at USTC. His research interests mainly in WSN and WBAN. His postal address is 96 JinZhai Road, Baohe District, Hefei, China.

Albert Y. Zomaya [F] is a Chair Professor and director of the Centre for Distributed and High Performance Computing at the University of Sydney. He has published more than 600 scientific papers and is an author, co-author, or editor of more than 20 books. He is the Editor-in-Chief of IEEE Transactions on Sustainable Computing and ACM Computing Surveys and serves as an Associate Editor for several leading journals. He is a Fellow of IEEE, AAAS, IET, and member of Academia European. His address is University of Sydney, Sydney, Australia.

Chunbo Luo received the Ph.D. degree in highperformance cooperative wireless networks from the University of Reading, Reading, U.K. His research has been supported by NSFC, Royal Society, EU H2020, and industries. His research interest focuses on developing model-based and machine learning algorithms to solve networking and engineering problems, such as wireless networks, with a particular focus on unmanned aerial vehicles. Dr. Luo is a Fellow of the Higher Education Academy.