

# Map2Schedule: An End-to-End Link Scheduling Method for Urban V2V Communications

Lihao Zhang, Haijian Sun, Jin Sun, Ramvijas Parasuraman, Yinghui Ye, Rose Qingyang Hu

**Abstract**—Urban vehicle-to-vehicle (V2V) link scheduling with shared spectrum is a challenging problem. Its main goal is to find the scheduling policy that can maximize system performance (usually the sum capacity of each link or their energy efficiency). Given that each link can experience interference from all other active links, the scheduling becomes a combinatorial integer programming problem and generally does not scale well with the number of V2V pairs. Moreover, link scheduling requires accurate channel state information (CSI), which is very difficult to estimate with good accuracy under high vehicle mobility. In this paper, we propose an end-to-end urban V2V link scheduling method called Map2Schedule, which can directly generate V2V scheduling policy from the city map and vehicle locations. Map2Schedule delivers comparable performance to the physical-model-based methods in urban settings while maintaining low computation complexity. This enhanced performance is achieved by machine learning (ML) technologies. Specifically, we first deploy the convolutional neural network (CNN) model to estimate the CSI from street layout and vehicle locations and then apply the graph embedding model for optimal scheduling policy. The results show that the proposed method can achieve high accuracy with much lower overhead and latency.

**Index Terms**—V2V, spectrum sharing, link scheduling, convolutional neural network, graph neural network.

## I. INTRODUCTION

As vehicles become increasingly intelligent, thanks to powerful onboard chips and sensors, it is becoming both inevitable and progressively feasible to enable them to establish connections with other vehicles (V2V), infrastructure (V2I), and pedestrians (V2P) for advanced transportation applications. This is the core concept of vehicle-to-everything (V2X) technologies [1], which aims to enhance the transportation system to become more comfortable, environment-friendly, efficient, and safer than ever. Among the various types of V2X communications, V2V plays a crucial role due to its potential to facilitate autonomous driving and provide timely incident alerts. Nevertheless, a significant portion of V2V communications will occur in dense urban environments, potentially resulting in substantial mutual interference and calling for appropriate link scheduling mechanisms.

To better utilize the wireless resources, a common practice is to divide the wireless channels into small resource blocks

L. Zhang and H. Sun are with the School of Electrical and Computer Engineering, University of Georgia, Athens, GA, USA.

J. Sun and R. Parasuraman are with the School of Computing, University of Georgia, Athens, GA, USA.

Y. Ye is with Shanxi Key Laboratory of Information Communication Network and Security, Xi'an University of Posts & Telecommunications, China.

R. Q. Hu is with the Electrical and Computer Engineering Department at Utah State University, Logan, UT, USA.

in either the time or frequency domain. While this can help mitigate interference, spectrum utilization may not be optimal and can potentially be further enhanced. Dynamic Spectrum sharing, on the other hand, allows multiple users to access the same spectrum resources. The active users are allowed to share the spectrum resources when their mutual interference is deemed low. As a result, many optimal scheduling algorithms have been developed using diverse mathematical frameworks. Nonetheless, these algorithms tend to suffer from the drawback of high computational complexity. For example, [2] proposed a global-optimal scheduling algorithm called *S-MAPEL*, which has a high complexity with respect to pairing the links.

To mitigate the concern of high computational complexity, certain sub-optimal algorithms have been developed. One of the early contributions in this area is the greedy heuristic search algorithm *FlashLinQ* [3]. Authors in [4] proposed a general framework that link scheduling is optimal for the whole generalized-dimension-of-freedom (GDoF) region and within a constant gap of the capacity region when it is subject to the treat interference-as-noise (TIN) condition. Based on TIN, *ITLinQ* [5] and *ITLinQ+* [6] were then developed to find the sub-optimal scheduling policy by sequential link selection. However, the performance results of *FlashLinQ*, *ITLinQ*, and *ITLinQ+* heavily depend on empirical parameter design. In [7], the authors proposed another iterative algorithm, *FPLinQ*, derived from fractional programming. Although this algorithm outperforms the above algorithms with no need to do parameter fine-tuning, its iterative process still reveals high computation complexity that can hardly meet real-time V2V communication requirements under high mobility.

Furthermore, it is worth noting that the above methods are based on the assumption that full CSI (including direct and interference channels) is available. The direct CSI for a transmitter-receiver pair can be acquired during their communication process. However, the link scheduling problem not only needs the direct CSI but also needs the information about all the interfering channels, a task that proves to be quite challenging and time-consuming in the high-mobility scenarios. With the surge in machine learning (ML) technologies, some end-to-end ML-based scheduling methods have been developed. In [8], the authors proposed “spatial deep learning”, an end-to-end ML approach that generates link scheduling based on spatial distances. In [9], the authors proposed a fast link scheduling based on the graph embedding model by treating the V2V links as graphs. However, the above schemes only considered transmitter-receiver pair distance during scheduling, which may

work well in rural, wild, or flat environments as distance has a strong correlation with CSI. Distance based scheduling can lead to significant scheduling policy deviations in urban V2V scenarios where buildings and foliage are dense.

To achieve real-time and optimal V2V link scheduling, it is essential to have both accurate full CSI estimation and low-complexity scheduling algorithms in place. Furthermore, it is crucial that the scheme needs to scale well with an increasing number of links. In this work, we explore a new approach driven by ML and present the following contributions.

- We develop an end-to-end link scheduling method named *Map2Schedule*, which can directly generate a scheduling policy that requires only city maps and vehicle locations, thereby significantly reducing signal overhead and computation complexity.
- *Map2Schedule* is trained on physical-model-based simulation data and performs 80% higher on sum-rate metrics compared to the existing methods such as [8] and [9]. Meanwhile, *Map2Schedule* exhibits low latency and can schedule 50 V2V links within 0.2 seconds, a reduction of two orders of magnitude compared to our performance baseline. This runtime was tested without any optimization applied to the computation process. Besides, the proposed algorithm can handle an arbitrary number of V2V links, which can greatly facilitate transfer learning and practical implementation.
- We have also explored transfer learning for better adaptation to unforeseen scenarios. The results show that our model has good transferability and can quickly adapt to new tasks with a few-shot method.

The rest of the paper is organized as follows. In section II, the end-to-end wireless link scheduling problem is presented and the problem is divided into two sub-problems. Section III introduces the dataset and the system model. In section IV, the experiment settings and the results are presented. Finally, this paper is concluded in section V.

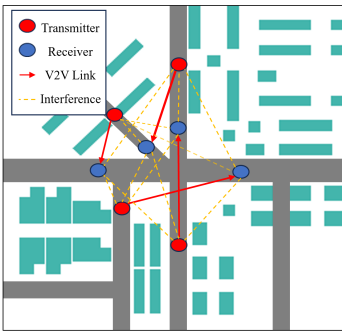


Fig. 1: A typical urban V2V scenario.

## II. BACKGROUND AND PROBLEM FORMULATION

As shown in the Fig. 1, we consider a typical urban V2V scenario which consists of a city map and  $N$  overlaid wireless links denoted as  $d_i \in \mathcal{D}\{d_1, d_2, d_3, \dots, d_N\}$ .

The gray areas are roads and the green ones are buildings. Each  $d_i$  refers to a pair of a transmitter  $t_i$  and its corresponding receiver  $r_i$ . Consequently, link set  $\mathcal{D}$  corresponds to  $N$  transmitters  $t_i \in \mathcal{T}\{t_1, t_2, t_3 \dots t_N\}$  and  $N$  receivers  $r_i \in \mathcal{R}\{r_1, r_2, r_3 \dots r_N\}$ . To boost spectrum efficiency, all the links share the whole frequency resource. We denote  $h_{ii}$  as the CSI from  $t_i$  to  $r_i$ , also is the direct channel gain of link  $d_i$ , and denote  $h_{ji}$  for CSI from  $t_j$  to  $r_i$ , also is the interference gain from link  $d_j$  to link  $d_i$ .

In the considered urban V2X scenario, we aim to find the optimal link scheduling policy (on or off state of each  $d_i$ ) to maximize the system sum-rate. In this paper, we only schedule the links to either be activated with power  $p_i$  or be turned off, which is represented by a binary indicator  $x_i \in \{0, 1\}$  for link  $d_i$ . Without loss of generality, we assume all links have the same power  $p_i$ .  $\mathbf{x} = [x_1, x_2, \dots, x_N]$  denotes the link scheduling policy of  $\mathcal{D}$ . The signal-to-interference plus noise ratio (SINR) of  $d_i$  can be written as Eq. (1) where the  $\sigma_N^2$  denotes the additive white Gaussian noise (AWGN), and the communication rate of  $d_i$  can be written as Eq. (2) where  $B$  is the system bandwidth.

$$SINR_i(\mathbf{x}) = \frac{x_i p_i |h_{ii}|^2}{\sigma_N^2 + \sum_{j \neq i} x_j p_j |h_{ji}|^2}, \quad (1)$$

$$R_i(\mathbf{x}) = B \log_2(1 + SINR_i(\mathbf{x})). \quad (2)$$

From Eq. (2), we could find that the communication rate of  $d_i$  is impacted by interference. The optimization problem can be formulated as (3), which means we should find a proper scheduling policy  $\mathbf{x}$  that only activates the ‘‘good’’ links with high channel gain and low interference to others.

$$\begin{aligned} \max_{\mathbf{x}} \quad & \sum_{i=1}^N R_i(\mathbf{x}) \\ \text{s.t.} \quad & x_i \in \{0, 1\}, \forall x_i \in \mathbf{x}. \end{aligned} \quad (3)$$

This problem naturally splits into two sub-problems. The first sub-problem is to obtain  $h_{ij}, \forall i, j \in \{1, 2 \dots N\}$ , the  $N \times N$  CSI values between all possible transmitter-receiver pairs. The second sub-problem is to generate the link scheduling policy based on the CSI values.

### A. Channel Estimation

In practice, CSI is obtained by periodically sending pilot signals. However, they experience high overhead and latency, not to mention implementing a centralized algorithm for indirect CSI pairs. In the literature, one of the most accurate approaches is physical-model-based simulations, like the dominant path model (DPM) [10] and ray tracing [11]. Although such simulations can provide precise CSI prediction, it will take minutes to generate full CSI. In the past few years, some ML-based CSI predicting methods emerged. However, most are developed for single transmitter-receiver pairs, like [12]–[14].

Such single-pair methods are not applicable to large-scale V2V cases, where the number of links can reach to hundreds. Fortunately, with recent advancements in ML research, new

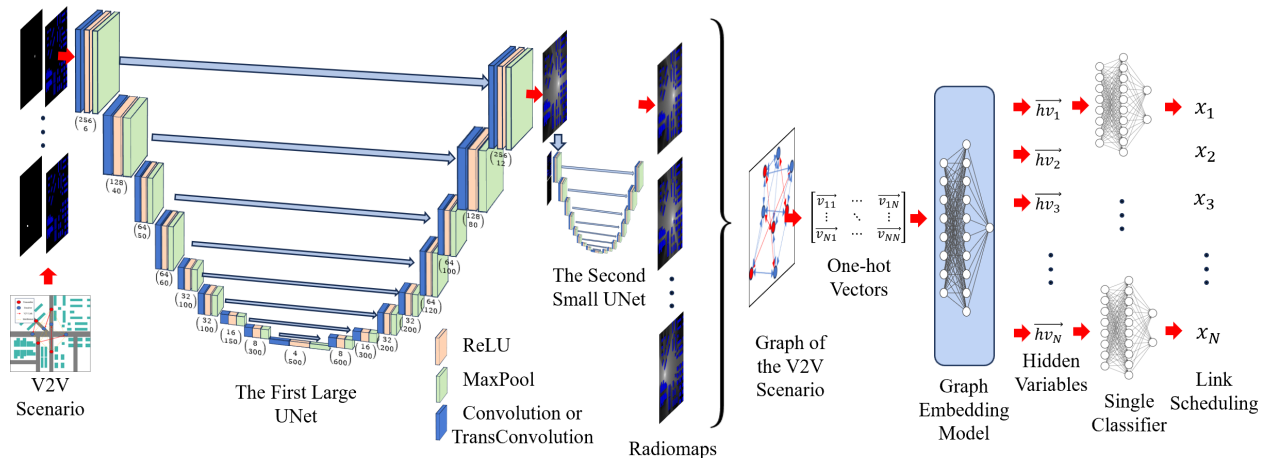


Fig. 2: Map2Schedule structure

deep learning methods were developed and can emulate the physical model with lower computation complexity.

### B. Link Scheduling

The second sub-problem is to find optimal or sub-optimal link scheduling policy based on available CSI. In fact, given all  $h_{ij}$ , Eq. (3) is a classical optimization problem. Traditional scheduling algorithms use various optimization techniques to find the optimal  $\mathbf{x}$ . As discussed in the introduction part, *S-MAPEL* [2], *FlashLinQ*, *ITLinQ*, *ITLinQ+*, and *FPLinQ* represent past decade’s efforts in this field. Nevertheless, they all suffer from high computation complexity and poor scalability, hence are not suitable for the considered V2V scenarios.

Recently, some ML-based scheduling algorithms with low computation complexity have been proposed. The closest works are [8] and [9]. Both works are end-to-end scheduling predictions and only take link distance as a system input. Their performance builds on the implicit correlation between distance and CSI and, therefore will not work well where distance and channel have large deviations, such as in urban environments. Besides, they can also suffer from poor generalizability, which hinders practical applications.

## III. SCHEDULING VIA MAP2SCHEDULE

We present the details of the proposed *Map2Schedule* approach, which is designed to generate scheduling policy  $\mathbf{x}$  directly from information provided in Fig. 1. The proposed system is illustrated in Fig. 2. The first component is a modified CNN model from *RadioUNet* [15], which predicts the CSI from the 2-dimensional (2-D) city map and vehicle locations. The second component is the graph embedding model, which uses the predicted CSI to generate a near-optimal scheduling policy in real time.

### A. Dataset

In this paper, our dataset consists *RadioMapSeer* [15] and the near-optimal scheduling policy  $\mathbf{x}$  generated by *FPLinQ*. *RadioMapSeer* is a dataset of 56,000 radiomaps. As shown

in Fig. 3, each radiomap is an image of the signal strength distribution, and a brighter pixel corresponds to a stronger signal at that location. These radiomaps were simulated on 700 2-D city maps. For each map, there are 80 radiomaps simulated by WinProp with the DPM algorithm, corresponding to 80 transmitter locations. The map size is 256 m  $\times$  256 m.

For our scheduling task, we built V2V scenarios from the *RadioMapSeer* dataset. These scenarios were configured based on two key parameters: the number of links and the range of link distances. The number of links is configured as 10, 20, 30, 40, and 50, and the range of link distance is configured as 2 meters to 32 meters for short-distance groups and 2 meters to 65 meters for long-distance ones. We extracted the CSI matrix from the radiomap and then utilized the *FPLinQ* algorithm to obtain link scheduling policy as the baseline (ground truth).

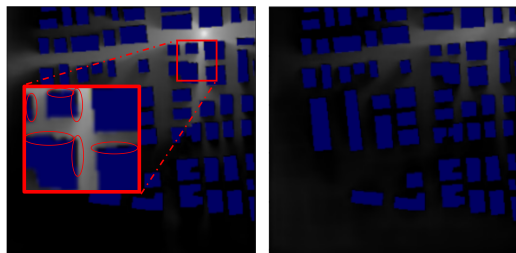


Fig. 3: Black edge and transmitter fading

### B. Radiomap Prediction

To predict the CSI, we deployed the *RadioUNet*, a CNN model based on the *UNet* architecture. *UNet* architecture is symmetric and with similar shape as letter “U” as shown in Fig. 2. This architecture consists of multiple levels, with two network blocks on each level. The left-side blocks perform convolution (down-sampling), activation, and pooling, generally used as feature extracting or encoding. On the right side, corresponding right-side blocks will do transposed convolution (up-sampling), activation, and pooling, acting as feature expanding or decoding. Unlike other CNN models, the prominent

feature of *UNet* is the encoder-decoder path on each level as the blue long arrows in Fig. 2. These paths will send the output of the left-side blocks into the input of the right-side blocks.

Furthermore, *RadioUNet* has two connected *UNet* architectures. Both of the *UNet* architectures have ten levels, but the first *UNet* has more channels than the second one. The inputs of the first *UNet* consist of one 2-D city map and one transmitter location image. Its output is a coarse radiomap prediction. After that, the inputs of the second one *UNet* are nearly identical to the first *UNet*, except the output of the first *UNet* is added.

Original *RadioUNet* is trained for just one transmitter. We have modified its input structure to support multiple direct pairs. Since all links use the same frequency,  $N$ -pair CSI can be obtained by stacking up each individual radiomap and fine-tuning the output layer. Besides, original *RadioUNet* exhibits two issues. The first issue was black edges as illustrated on left side of Fig. 3, where noticeable black edges surround the “bright faces” of buildings. These black edges correspond to mispredicted zero channel gain at that location. One conjecture is that the model has learned the pattern of black edges associated with “dark faces” of buildings and mistakenly applied this knowledge to “bright faces” during prediction. To rectify this issue, a straightforward solution is to detect abrupt changes in non-building areas and replace the abnormal pixels with their largest neighbors.

The second issue is “transmitter fading” as shown on the right side of Fig. 3. The transmitters in edge areas significantly “fades”. This issue can be attributed to biased data in the training set, where there are few transmitters located at the map edges. Methods to mitigate data bias include adding compensatory data and data augmentation, such as segmenting the original radiomaps to generate additional samples with transmitters at the edges. In our implementation, we addressed this issue by dropping transmitters at selected locations.

### C. Graph Embedding Link Scheduling

Essentially, the link scheduling is to determine the relationship between the direct CSI ( $h_{ii}$ ) and the interference to all other receiver nodes ( $h_{ij}, j \neq i$ ). Such a model can naturally connect with graph structure. In this paper, we applied and modified a popular graph embedding model *structure2vec* [16], which is a powerful method for data with graph structure. Its main idea is constructing embedded hidden variables via a nonlinear learnable function for each graph data point (each node in the graph). These hidden variables represent the whole graph from the perspective of each node, which means these variables encapsulate information about the node itself and its relationship to the whole graph. Depending on the problem objectives, these embedded hidden variables could be processed by classifiers, regression models, or other appropriate models. After determining the whole structure, the embedding learnable function and the subsequent task model will be optimized simultaneously through each backward propagation process.

To implement the graph embedding model on link scheduling tasks, the first step is to represent each V2V scenario as a graph. While it might seem intuitive to treat each transmitter

and receiver as a node, doing so will result in each node missing its node feature. This is unsuitable for graph embedding models that highly depend on the node feature to embed. Considering that our task is to find “good” links, treating each link as a node is a more natural idea and will make it easier to design the subsequent classifier.

Denote  $G(V, E, \alpha)$  as the directed graph of all the links in one V2V scenario,  $v_i \in V$  denotes the node  $v_i$  in the nodes set  $V$ , which represents the link  $d_i$ .  $e_{ij} \in E$  is the edge from node  $v_i$  to node  $v_j$ .  $\alpha_{ij}$  denotes the edge feature of  $e_{ij}$  which is equal to  $h_{ij}$ , and  $\alpha_{ii}$  denotes the node feature of  $v_i$  which is equal to  $h_{ii}$ . From the graph embedding theory, the embedded hidden variable  $\mu_i$  for node  $v_i$  will be updated iteratively as Eq. (4).  $N(v_i)$  is the set of neighbors of  $v_i$ .

$$\mu_i^{(t+1)} = \Gamma(\alpha_{ii}, \{\alpha_{ji}\}_{j|v_j \in N(v_i)}, \{\mu_{v_j}^{(t)}\}_{v_j \in N(v_i)}). \quad (4)$$

Here,  $\Gamma$  represents the nonlinear learnable function for inferring the hidden variables, and there are several approximate inference algorithms. In this paper, we choose the mean-field inference algorithm, as shown in Eq. (5).

$$\mu_i^{(t+1)} = \sigma(W_1 \alpha_{ii} + W_2 \sum_{j|v_j \in N(v_i)} \alpha_{ji} + W_3 \sum_{j|v_j \in N(v_i)} \mu_j^{(t)}). \quad (5)$$

To facilitate model computation, the CSI is quantized to  $p$ -dimensional one-hot code. This quantization process compresses the continuous CSI feature space into discrete  $p$  categories. Since we aim to perform binary classification on each link, the  $p$  categories of CSI should both provide sufficient accuracy and enable faster link scheduling learning. In this paper, the node and edge features are embedded into 8-dimensional one-hot codes, the hidden variable are iterated twice, and the embedded hidden variables are represented as 32-dimensional vectors. Consequently, the shapes of  $W_1$  and  $W_2$  in Eq. (5) are both  $32 \times 8$ , while  $W_3$  is  $32 \times 32$ . The above setting of hyperparameters was found to be optimal through experiments in [9].

After the graph embedding model is constructed, the classifier for each hidden variable can be built. The classifier contains just one hidden variable layer. So, the size of the first weight is  $32 \times 64$ , and the second is  $64 \times 2$ . It may appear counter-intuitive that the classifier only processes one hidden variable at a time, lacking a global view. However, from Eq. (5), we can find that each embedded hidden variable is calculated from the whole graph, and the CSI values are formulated similarly to *FPLinQ*, which means those hidden variables should be able to mimic how *FPLinQ* algorithm does link scheduling. Meanwhile, the single classifier, as well as the previous embedding model, can accommodate any number of inputs. Hence, the proposed system architecture can accommodate an arbitrary number of V2V links without requiring any modifications. This capability offers significant advantages for practical implementation, transfer learning, and more.

### D. Training Process

The proposed system is trained as two interconnected components in a supervised fashion. For the first component, the

CNN model is trained using the *RadioMapSeer* dataset. As mentioned before, the 56,000 radiomaps were simulated on 700 city maps, with 80 different transmitter locations on each map. We divide the radiomaps based on different city maps, allocating them as follows: 400 for the training set, 100 for the validation set, and 200 for the test set. The mean square error (MSE) loss function is employed and the Adam optimization algorithm is used with a learning rate of  $10^{-4}$ .

In the second component, the training input consists of V2V CSI extracted from DPM-simulated radiomaps, and the target link scheduling policy is generated by the *FPLinQ* algorithm, serving as the ground truth. The dataset splits follows 800/1000/4000 for train/validation/test respectively. The reason for such a small training set is as follows. The work in [9] shows that training on a small dataset has a similar performance compared with large datasets.

Moreover, our problem involves a smaller map size, the links cluster and more overlap in these scenarios, which lead to low link activation ratio and make our problem more challenging. Due to the low activation ratio, the original graph embedding model can not learn the pattern quickly and precisely, as shown in the result section. For example, the prediction of sum-rate can only achieve about 60% of ground truth *FPLinQ* performance. This inconsistency comes from the biased data of much more “bad” links in each V2V scenario. Because of the utilization of the normal binary cross entropy (BCE) loss function, the model will focus on all input data samples equivalently and will be trained more by the “bad” link samples. Consequently, the trained model will have low accuracy on “good” links, which is more important to the sum-rate performance in our scenarios.

To alleviate the bias, we deployed the weighted cross entropy loss function. By adjusting the weight of the classes, the accuracy of good links weighs more in the loss function. It leads the parameters of the model descent along the direction that improves the accuracy on good links and the sum-rate metric. In addition, given the variations in the CSI distribution caused by different link numbers and ranges of link distance, we conducted the grid search for optimal training hyperparameters on each scenario setting.

#### IV. EXPERIMENTAL RESULTS

Here, we present the results of our experiments. The training process and test experiments were implemented in PyTorch.

##### A. Sum-rate Performance

As mentioned before, we choose *DPM* simulation for getting accurate CSI and then use *FPLinQ* algorithm for near-optimal scheduling as the 100% performance baseline (DPM-FP). In Fig. 4, “M2S 2m-32m” and “M2S 2m-65m” represent the sum-rate performance of *Map2Schedule* under short ( $d_i$  from 2m to 32m) and long ( $d_i$  from 2m to 65m) link distance, respectively. The “DF-FP 2m-32m” and “DF-FP 2m-65m” represent the performance of the distance-based fading model (with only vehicle locations) and the *FPLinQ* (DF-FP), which is the 100% performance baseline of paper [7], [9]. The distance-based

fading model applied the short-range outdoor model ITU-1411 with a distance dependent path-loss. *Map2Schedule* can achieve over 90% baseline performance on sum-rate metrics. However, the DF-FP method can only achieve approximately 50%. This verified that street layout together with vehicle locations in our model, can predict almost optimal scheduling policy while distance alone cannot provide satisfactory performance.

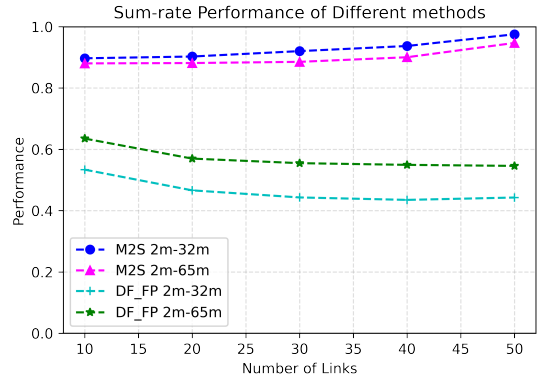


Fig. 4: Sum-rate performance

TABLE I: Impacts of weighted BCE loss function

Method	Map2Schedule	Original Graph Embedding Model
Average accuracy	79.08%	<b>82.35%</b>
Average sum-rate	<b>91.27%</b>	67.29%
Sum-rate in 50-long scenarios	<b>94.71%</b>	53.83%

##### B. Impacts of Weighted BCE Loss Function

We compared the scheduling performance of the original graph embedding model and the second part of *Map2Schedule* on the same CSI matrix input. As mentioned above, we deployed the weighted BCE loss function to address the scenario bias. We compare two metrics: average accuracy, which is the number of correctly predicted  $x_i$  over  $N$ , and average sum-rate, which is the ratio of the objective function in Eq. (3) given predicted  $x_i$  to DPM-FP. The result in Table I reveals a slight reduction in average accuracy when using the weighted BCE loss function. This reduction can be attributed to the modification of the model, which makes the model focus more on “good” link (high data rate) samples. Therefore, the accuracy on “bad” link (low data rate) samples decreased, which contributed more to the total accuracy. However, the sum-rate metric significantly depends on the accuracy of “good” links when “good” links constitute a small proportion. As a result, *Map2Schedule* with weighted BCE loss function was improved by an average of 35% on the sum-rate metric and 76% on the sum-rate metric in scenarios with 50 pair of longer links.

##### C. Transferability between Different Scenarios

Transfer learning is a widely used ML technology that aims to improve the ML model performance on the target domain by transferring the knowledge from the original domain. In

V2V scenarios, the vehicle density can change significantly within a single day, leading to very different CSI patterns. In this section, we tested our long-distance group model, which is the original domain knowledge, with the short-distance group data, which is the target domain. “Zero-shot” refers to without any view of the target domains, and “few-shot” refers to only viewing a few samples of the target domains. The “non-transfer” refers to the performance of models completely trained on the target domains. From the performance in Fig. 5, we can find that our model exhibited promising transferability across different scenarios. However, the performance of transferred knowledge is lower than the original domain knowledge. The few shots slightly improved the performance, but further research on transfer learning is still needed for practical implementation.

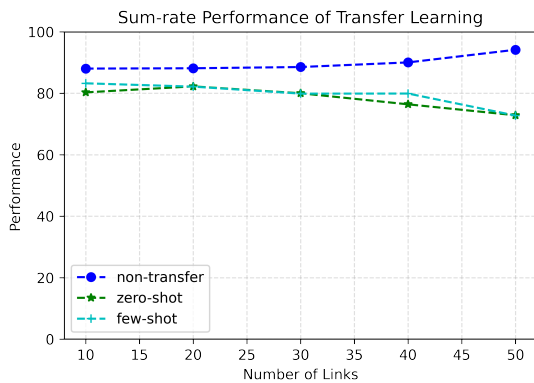


Fig. 5: Sum-rate performance of transfer learning

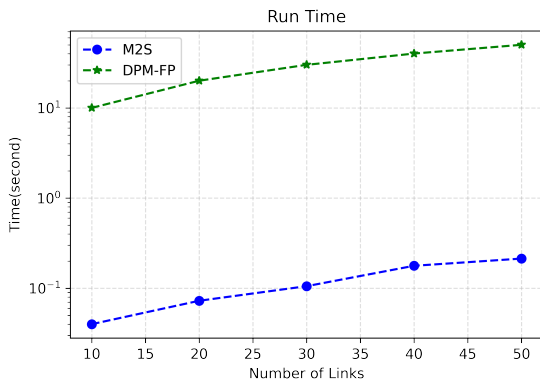


Fig. 6: Run time comparison with baseline method

#### D. Run Time and the Computation Complexity

All the experiments were conducted on our desktop with AMD Ryzen 5955WX processor and NVIDIA RTX A6000 graphic card. The average run time is shown in Fig. 6. Specifically, the run time of DPM-FP is on the order of  $10^1$  seconds, and the run time of *Map2Schedule* is on the order of  $10^{-1}$  second. Besides, further improvements can be achieved by carefully handling quantization in the data flow of embedding model. This result indicates *Map2Schedule* has the desired low computational complexity and real-time feature.

## V. CONCLUSION

In this paper, we proposed an ML-based end-to-end wireless link scheduling approach called *Map2Schedule*, which is specifically designed for urban V2V scenarios. *Map2Schedule* can generate near-optimal link scheduling policy from vehicle locations and the city map. In urban environment, our approach remarkably outperforms those distance-based scheduling methods and competes with the state-of-the-art physical-model-based ones. Meanwhile, *Map2Schedule* requires little computation resources, allowing it to meet the real-time requirement of V2V communications.

## REFERENCES

- [1] S. Chen, J. Hu, Y. Shi, Y. Peng, J. Fang, R. Zhao, and L. Zhao, “Vehicle-to-everything (v2x) services supported by lte-based systems and 5g,” *IEEE Communications Standards Magazine*, vol. 1, no. 2, pp. 70–76, 2017.
- [2] L. P. Qian and Y. J. Zhang, “S-mapel: Monotonic optimization for non-convex joint power control and scheduling problems,” *IEEE Transactions on Wireless Communications*, vol. 9, no. 5, pp. 1708–1719, 2010.
- [3] X. Wu, S. Tavildar, S. Shakkottai, T. Richardson, J. Li, R. Laroya, and A. Jovicic, “Flashling: A synchronous distributed scheduler for peer-to-peer ad hoc networks,” *IEEE/ACM Transactions on networking*, vol. 21, no. 4, pp. 1215–1228, 2013.
- [4] C. Geng, N. Naderializadeh, A. S. Avestimehr, and S. A. Jafar, “On the optimality of treating interference as noise,” *IEEE Transactions on Information Theory*, vol. 61, no. 4, pp. 1753–1767, 2015.
- [5] N. Naderializadeh and A. S. Avestimehr, “Itling: A new approach for spectrum sharing in device-to-device communication systems,” *IEEE journal on selected areas in communications*, vol. 32, no. 6, pp. 1139–1151, 2014.
- [6] X. Yi and G. Caire, “Optimality of treating interference as noise: A combinatorial perspective,” *IEEE Transactions on Information Theory*, vol. 62, no. 8, pp. 4654–4673, 2016.
- [7] K. Shen and W. Yu, “Fpling: A cooperative spectrum sharing strategy for device-to-device communications,” in *2017 IEEE international symposium on information theory (ISIT)*. IEEE, 2017, pp. 2323–2327.
- [8] W. Cui, K. Shen, and W. Yu, “Spatial deep learning for wireless scheduling,” *IEEE journal on selected areas in communications*, vol. 37, no. 6, pp. 1248–1261, 2019.
- [9] M. Lee, G. Yu, and G. Y. Li, “Graph embedding-based wireless link scheduling with few training samples,” *IEEE Transactions on Wireless Communications*, vol. 20, no. 4, pp. 2282–2294, 2020.
- [10] R. Wahl, G. Wölfle, P. Wertz, P. Wildbolz, and F. Landstorfer, “Dominant path prediction model for urban scenarios,” *14th IST Mobile and Wireless Communications Summit, Dresden (Germany)*, 2005.
- [11] K. Rizk, J.-F. Wagen, and F. Gardiol, “Two-dimensional ray-tracing modeling for propagation prediction in microcellular environments,” *IEEE Transactions on Vehicular Technology*, vol. 46, no. 2, pp. 508–518, 1997.
- [12] K. Saito, Y. Jin, C. Kang, J.-i. Takada, and J.-S. Leu, “Two-step path loss prediction by artificial neural network for wireless service area planning,” *IEICE Communications Express*, vol. 8, no. 12, pp. 611–616, 2019.
- [13] S. P. Sotiroudis, S. K. Goudos, K. A. Gotsis, K. Siakavara, and J. N. Sahalos, “Application of a composite differential evolution algorithm in optimal neural network design for propagation path-loss prediction in mobile communication systems,” *IEEE Antennas and Wireless Propagation Letters*, vol. 12, pp. 364–367, 2013.
- [14] T. Imai, K. Kitao, and M. Inomata, “Radio propagation prediction model using convolutional neural networks by deep learning,” in *2019 13th European Conference on Antennas and Propagation (EuCAP)*. IEEE, 2019, pp. 1–5.
- [15] R. Levie, Ç. Yapar, G. Kutyniok, and G. Caire, “Radiounet: Fast radio map estimation with convolutional neural networks,” *IEEE Transactions on Wireless Communications*, vol. 20, no. 6, pp. 4001–4015, 2021.
- [16] H. Dai, B. Dai, and L. Song, “Discriminative embeddings of latent variable models for structured data,” in *International conference on machine learning*. PMLR, 2016, pp. 2702–2711.