XRouting: Explainable Vehicle Rerouting for Urban Road Congestion Avoidance using Deep Reinforcement Learning

Zheng Wang
Beijing-Dublin International College
University College Dublin
Belfield, Dublin, Ireland.
zheng.wang@ucdconnect.ie

Shen Wang
School of Computer Science
University College Dublin
Belfield, Dublin, Ireland
shen.wang@ucd.ie

2022 IEEE International Smart Cities Conference (ISC2) | 978-1-6654-8561-6/22/$31.00 ©2022 IEEE | DOI: 10.1109/ISC255366.2022.9922404

***Abstract*—Rerouting vehicles for urban congestion avoidance is challenging as the decision has to be undertaken promptly with the consideration of traffic condition changes caused by other vehicles' routing plans. Existing solutions such as the on-board navigation systems (e.g., Google Maps) cannot meet these requirements which is prone to trigger the well-known routing oscillation problem. Though deep reinforcement learning (DRL) approaches are able to provide a high-quality solution and satisfy the real-time requirement, not only do they usually suffer the slow and instability issues for convergence, but the input information, like a picture for each time step, is also teeming with redundant information. In this paper, we propose XRouting model that uses policy-based DRL and the revised Gated Transformer (GTr) architecture to accelerate and stabilize the training convergence in solving dynamic routing problems. Our simulation study validates that compared with existing rerouting solutions, XRouting can achieve higher reductions in travel time, fuel consumption, CO_2 emission, and the route length. More importantly, XRouting is capable of determining which features are predominant when vehicles conduct rerouting. This explainable ability of our model can further guide human drivers what features to consider when rerouting manually in real life.**

I. INTRODUCTION

Drivers often need to change their pre-planned routes to avoid potential congestion due to the changing urban road traffic condition. It is a challenging decision-making process as an effective vehicle re-routing often requires a timely response with high-frequency traffic condition prediction. A timely response can ensure that any suggested re-routing plan can be well taken before missing chances when vehicles pass through incoming intersections. The high-frequency traffic prediction can help mitigate the potential routing oscillation problem as shown in Fig.1. However, such a high-quality prediction demands high computation (i.e., compute a collection of optimal routes for multiple vehicles) and communication (i.e., to exchange routing decisions of surrounding vehicles) costs thus it cannot be provided promptly. Similarly, if we impose the short time limit, the re-routing solution will likely not be effective.

Traditional methods such as the onboard navigation systems (e.g., Google Maps) use the shortest pathfinding algorithms (e.g., Dijkstra's algorithm) to update the route with the current traffic conditions. Due to the lack of consideration of other vehicles' re-routing decisions and therefore the near future traffic condition, research has demonstrated that with the increasing penetration rate of this system, the re-routing effectiveness gets worse. Recent methods such as next road re-routing [1] rely on the heuristic function which could lead to an unbounded worst solution when applied in different scenarios.

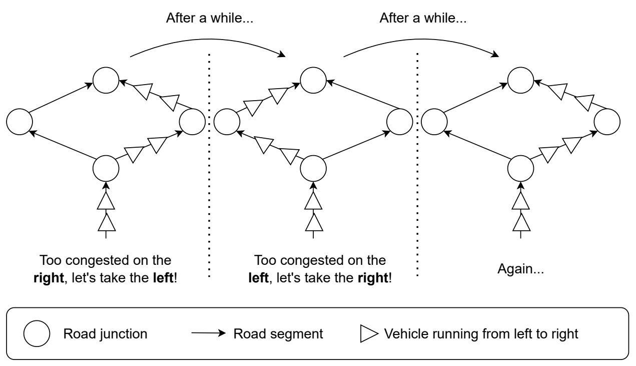
After a while...
After a while...
Too congested on the **right**, let's take the **left**!
Too congested on the **left**, let's take the **right**!
Again...
Road junction
Road segment
Vehicle running from left to right

Fig. 1. An illustration of routing oscillation problem. This shows an inappropriate re-routing strategy is likely to cause a long-term congestion due to routing oscillation.

Deep reinforcement learning (DRL) approaches have recently shown substantial potential in solving problems that are computationally intractable such as AlphaGo [2]. Koh et al. [3] and Geng et al. [4] also tried to utilize the deep Q network (DQN) model for vehicle re-routing to avoid road congestions. However, this approach has only been applied to one vehicle under light traffic conditions. Therefore it is still unclear if this solution has positive implications for alleviate the routing oscillation issue.

Furthermore, transformer has been exhibiting extraordinary talents in both natural language processing problems [5], [6] and computer vision problems [7] in recent years. In virtue of the excellent performance of transformer in dealing with the sequential problem, the attention mechanism used in its encoder-decoder structure is also exploited in reinforcement learning to solve NP-hard optimization problems including Travelling Salesman Problems (TSP) [8], and Vehicle Routing Problems (VRP) [9]. However, most of the previous transformer-based DRL model used in NP-hard optimization problems concentrates on nodes' information rather than edges, which is distinguished from the real world scenario of vehicle rerouting problem. The most decisive information of the latter one is usually provided by current roads rather than intersections. Moreover, it is common that there are multiple roads connecting to a same intersection in the real world, which encourages us to utilize the roads' information for solving real-time rerouting problems.

Then, in this paper, we propose a new DRL-based re-routing system called XRouting to fill the aforementioned gaps. Our contributions are summarized as follows:

- **Firstly, XRouting has a more stable convergence with less training time.** Our model regards all the roads as an ordered sequence rather than mixing all the features together or a picture with enormous redundant information. Compared with the state-of-the-art DQN model, our approach uses policy-based DRL - Proximal Policy Optimization (PPO) [10] and the revised Transformer architecture [11] to obtain a faster and more stable convergence in training. Besides, unlike the traditional routing problem [12] which usually neglects the relative position information of the input observation, our model incorporates position encoding component by interpreting the distance between the current position of the vehicle and the destination.
- **Secondly, the effectiveness of XRouting has been validated in a more realistic scenario.** Compared with the other competing DRL approach [3], our testing environment has 10 more roads and 100 more traffic trips in average. Instead of single lane for each road, our scenario defines two lanes for each road. Moreover, we also include the impact that traffic lights have on routing decision process. Besides, XRouting is evaluated under multiple vehicles showing the high robustness to the well-known routing oscillation problem, rather than just evaluating one single vehicle.
- **Thirdly, the explainability of XRouting is also demonstrated with the decomposition of the routing process.** Specifically, we propose a general method to visualize the feature values under each automated decision process. We have sampled approximately 2000 cases in average to reveal that when choosing the next turn among four or three choices, road's average speed and vehicle density are the most two significant factors to be considered by DRL agents. This result is particularly useful, because the human drivers are capable of learning from the simulation before XRouting puts into practice. Last but not least, we have shared our implementation of XRouting at[1] for others to replicate.

[1]https://github.com/ZKBig/XRouting

II. Related Works

Vehicle rerouting occurs normally when traffic is changing unexpectedly. Pan et al. [13] proposed five different rerouting strategies based on the traditional shortest pathfinding algorithm with traffic predictions. However, there is no clear winner among those five rerouting strategies. Moreover, with the growth of penetration rate, those rerouting methods get worse, which hinders potential large-scale applications. A similar trend was found in [1]. Besides, NRR proposed in [1] is based on a heuristic solution which is likely to cause unbounded worse solutions when applied in unknown urban areas.

The DRL approach can provide high-quality solutions even when the problem was computationally intractable such as AlphaGo [2]. Koh et al. [3] starts to use value-based DQN to reroute vehicles. Another similar DRL approach [4] is also used for improving pedestrian traffic. However, those attempts are in a relatively early stage and are tested in oversimplified scenarios with only one agent applied. Explainable multi agent path finding has been demonstrated in a recent work [14] for collision avoidance. There is a need to extend explainable AI to vehicle rerouting for guiding human drivers to avoid en-route congestion.

The XRouting proposed in this paper tries to fill the gaps mentioned above to improve the practicability and effectiveness of rerouting vehicles for congestion avoidance with more stability and explainable ability.

III. Proposed Method

This section elaborates the design of our proposed XRouting including the modelling of state, action, and reward, the selection of the PPO algorithm, and the adaption of Transformer architecture to the vehicle re-routing problem.

A. State, Action, and Reward

The agent in our model is a single vehicle. In the training stage, XRouting trains the policy function of a single vehicle that learns how to choose the next road given a certain state in the environment. The environment is indeed the urban road network with changing traffic. In the evaluation stage, the trained policy function is applied to multiple vehicles to test its effectiveness in congestion avoidance. Specifically, the design of state, action and reward are described as follows.

State: The observation space defined in our model incorporates the following two parts:

- **Road Attributes**: The attributes of each road can be separated into dynamic part and static part. The former one contains three elements including the average speed of each road v_{avr}, the vehicle density (the number of vehicles divided by the road length) of each road d_{veh}, and the average travel time (the road length divided by the average vehicle speed v_{avr}) of each road t_{avr}. The latter one represents the inherent geographical information including the Euclidean distance between the start point of each road and that of the destination road, which is denoted by dis_{aim}, as the green line depicts in the Fig 2. Another static attribute is used to distinguish the destination road and other roads in virtue of one-hot encoding. The value of the destination road is 1 while other roads are set to 0. Consequently, the road attributes can be described as a five-tuple: $[v_{avr}, d_{veh}, t_{avr}, dis_{aim}, 1/0]$.

- **Vehicle-Road Attributes**: The Vehicle-Road Attributes imply the relative position information between the DRL vehicle and each road. To be specific, it is comprised of two parts: the first part represents the real-time distance between the current DRL vehicle and the destination road crossing certain road, which is given by
$$dis_{veh\_aim} = dis_{veh\_road} + l_{road} + dis_{road\_aim}, \quad (1)$$
where dis_{veh_road} is the Euclidean distance between the position of the DRL vehicle and the start position of a certain road; l_{road} is the length of the corresponding road to be crossed; dis_{road_aim} represents the distance between the end position of the road and the start position of the destination road. Therefore, this variable demonstrates the difference of relative distances between the DRL vehicle and the destination road by going across different roads. The red line in the Fig 2 demonstrates the defined dis_{veh_aim} if the DRL vehicle goes across road DC in the future. The second element conveying position information is angles between the DRL vehicle and each road, which is shown as the angle α in Fig 2. In a nutshell, the combination of relative distances and angle values is capable of depicting the dynamic position information between the DRL vehicle and each road.

Action: Taking the aim into account that encouraging the DRL vehicle to make a feasible choice of the next road with the consideration of the current traffic condition, the corresponding action space is, then, intuitively defined as the direction of next turning. However, the size of the action space is intimately associated with the degree of each intersection, meaning that the number of action choices for each intersection is essentially different. Therefore, the action space is defined as the maximum degree value among all intersections in the network. According to Fig 2, it can be clearly spotted that the red DRL vehicle has four choices of next roads, which are denoted by the numbers ranging from 0 to 3.

Reward: The training motivation is to make the DRL vehicle choose the next road with the minimum travelling time under the current network circumstance. It is not hard to envisage that the minimum travelling time is not equivalent to the minimum road length. Therefore, the corresponding reward of our model is intuitively given by:
$$r = T_{start} - T_{end}, \quad (2)$$
This reward r refers to the time difference between the vehicle stepping into the road T_{start} and leaving the road T_{end}, which is negative. Then, the training process is literally regarded as the processing of decreasing the travelling time of DRL vehicle crossing each road.

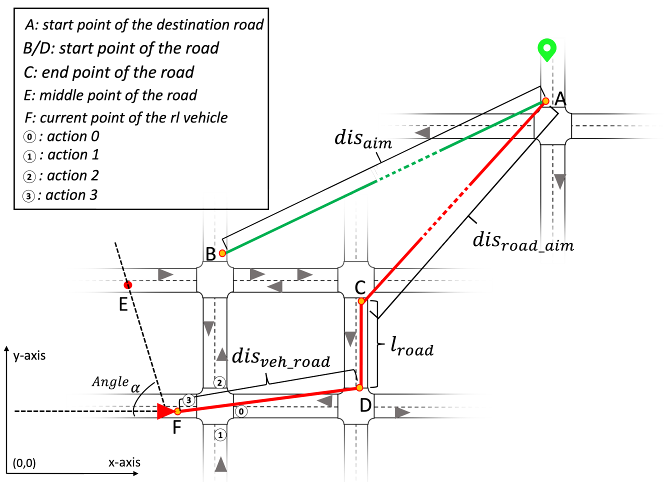

Fig. 2. The illustration of the state and action of XRouting. The red triangle is the DRL agent, while the gray one is the non-DRL vehicles.

B. Policy-based DRL

Our XRouting uses a policy-based DRL PPO algorithm because not only does the trust region method used in PPO ensure the policy update stability, but also the importance sampling of PPO provides an adequate sampling rate in the training process. The existing rerouting approaches use other value-based methods which have instability problems and are difficult for convergence in solving vehicle rerouting problems, especially when the testing scenario becomes much sophisticated. Moreover, from the engineering perspective, the PPO algorithm has the advantage of parameter tuning with relatively high efficiency. Based on the above comparison, PPO is universally used in various DRL problems with brilliant and stable performance. Then, the following contents introduce how XRouting uses PPO in detail.

Assume the route found by DRL vehicle is τ, then the discounted return of each route τ should be defined as $G(\tau) = \sum_{t=0}^{T-1} \gamma^t r_{t+1}$. The goal of our model is to find a policy function that can maximize the total discounted return (reward) of the process of DRL vehicle finding the complete route, which can be expressed in the form of expectation as:
$$\pi^* = argmax_{\pi} \sum_{\tau} (\sum_{t=0}^{T-1} \gamma^t r_{t+1}) P(\tau), \quad (3)$$
where π^* represents the optimal policy function, and $P(\tau)$ is the trajectory probability of routing generating processes, which is defined as $P(\tau) = P(s_0) \prod_{t=0}^{T-1} \pi(a_t, s_t) P(s_{t+1}|a_t.s_t)$. Based on that, the state value function can be further defined as:
$$V^{\pi}(s_t) = \mathbb{E}_{\pi}[\sum_{t=0}^{T-1} \gamma^t r_{t+1} | s_t] \quad (4)$$

In DRL, the policy function can be approximated by a neural network with parameter θ denoted by $\pi(a|s;\theta)$. Hence, in order to get the optimal policy function, policy gradient (PG) has been utilized sweepingly in virtue of gradient ascend to update the parameter θ. Then, the policy gradient with advantage function is given as:

$$\nabla J(\theta) = \mathbb{E}_{\pi_\theta}[\nabla log\pi(a_t|s_t;\theta)A^{\pi_\theta}(a_t,s_t)], \quad (5)$$

where the advantage function A^{π_θ} represents the advantage of taking action a_t in the current state s_t, which is calculated with the help of the state value function defined in equation (4). Moreover, equation (5) can be further modified by introducing importance sampling, which can be expressed as:

$$\nabla J(\theta) = \mathbb{E}_{\pi_{\theta'}}[\frac{\pi(a_t|s_t;\theta)}{\pi(a_t|s_t;\theta')}\nabla log\pi(a_t|s_t;\theta)A^{\pi_{\theta'}}(a_t,s_t)], \quad (6)$$

The core idea of this modification is to update policy function π^{θ} with the help of the old policy function $\pi^{\theta'}$. To avoid the large deviation between the two policy functions, trust region method with clipping is introduced given as:

$$\begin{aligned} L(\theta) = \mathbb{E}_\pi[min(&\frac{\pi(a_t|s_t;\theta)}{\pi(a_t|s_t;\theta')}A^{\pi_{\theta'}}(a_t,s_t), \\ clip(&\frac{\pi(a_t|s_t;\theta)}{\pi(a_t|s_t;\theta')}, 1-\epsilon, 1+\epsilon)A^{\pi_{\theta'}}(a_t,s_t))] \end{aligned} \quad (7)$$

The hyper parameter ϵ is used to control the size of the policy updating range, which enhances the stability of training process.

C. Revised Gated Transformer (GTr) architecture

Based on the above analysis, it is clear that two functions should be estimated in virtue of neural networks, that is, the policy function and the state value function. In our model, both those two functions are represented by the same neural network whose structure is shown in Fig. 3. As shown in Fig. 3, the proposed model is based on attention mechanism, in which the predefined attributes of all roads in the traffic network are the input sequence denoted by E. Note that for each road, the input attributes contain six elements: $[v_{avr}, d_{veh}, t_{avr}, dis_{aim}, angle_{veh}, 0/1]$, which represents the feature of each road and the relationship with the vehicle as aforementioned. The attribute dis_{veh_aim} is regarded as the input of the relative position embedding, which is denoted by P in our model. P implies the relatively priority of each road, with the help of the distance value, when it comes to the next road chosen for the rl-vehicle. Another equally important thing to note is that the input road sequence should be ascending ordered according to the relative position encoding to demonstrate the relative position difference of each road. The dim of E is $[1 \times 6 \times m]$, and the dim of P is $[1 \times m]$, where m is the total road number.

To improve the stability of training, $GRUGateLayer$ is used here by replacing the traditional residual connections [11]. Moreover, two normalization layers are added: one is before the MHA layer, the other is before the MLP forwarding layer. Then, the mathematical expression of the defined policy/value network in Fig. 3 can be written as follows:

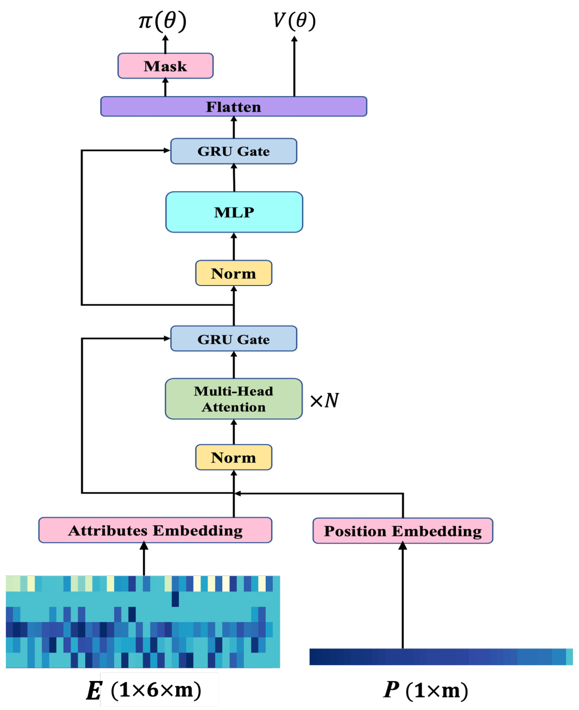

Fig. 3. The revised Transformer architecture of XRouting, where E represents the ordered input road attributes sequence, and P is the distance between the DRL vehicle and the destination with DRL vehicle crossing each road.

$$X = Concat[Embedding(E), Embedding(P)] \quad (8)$$

$$\widetilde{X} = MHA(LayerNorm(X)) \quad (9)$$

$$Y = GRUGate(X, X + \widetilde{X}) \quad (10)$$

$$\widetilde{Y} = MLP(LayerNorm(Y)) \quad (11)$$

$$Z = GRUGate(Y, Y + \widetilde{Y}) \quad (12)$$

$$\widetilde{Z} = Flatten(Z) \quad (13)$$

$$\pi(\theta) = Mask(Dense(\widetilde{Z})) \qquad V(\theta) = Dense(\widetilde{Z}) \quad (14)$$

Note that, the mask is applied to the generated policy probability results which have the dim of the maximum edge choices in the overall network, considering the different number of choices for each intersection. In other words, the probability values of the redundant choices will be converted to negative infinity, making them impossible to be chosen by the DRL vehicle.

IV. Experiments Design

A. Simulation settings

Software and Hardware: The traffic simulation platform used in our experiments is Simulation of Urban Mobility(SUMO) 1.13.0. The reinforcement learning training tool used in our experiment is RLlib 1.12.0. The traffic network environment for training is created based on OpenAI Gym.

Algorithm 1 PPO for XRouting

```
Initial the parameter of policy/value network $\theta$;
for iter = 0, 1, 2,... do
    for episode = 1, 2,... M do
        Reset traffic trips;
        Run policy $\pi_{\theta'}$ for T time steps and record the corresponding states and actions;
        Compute rewards $r_t$ for each time step;
        Compute advantage estimates $A^{\pi_{\theta'}}$ with the help of $V_{\pi_{\theta'}}$;
    end
    Update the parameter of policy/value network by maximizing the objective function $L(\theta)$.
end
```

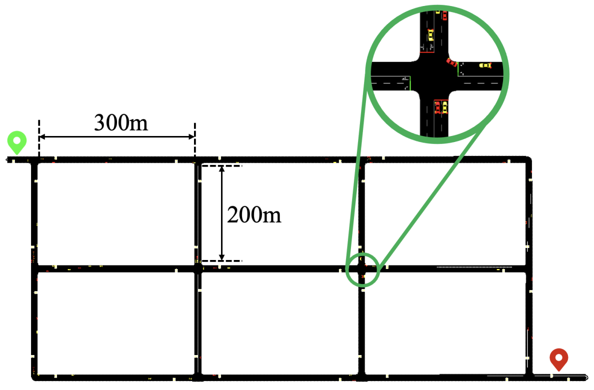

Fig. 4. Training Scenario

The CPU used for training is Intel Core i5 with 1.4 GHz. The main memory is 8GB.

Scenario: The scenario used in our experiment is a customized Manhattan-like traffic network shown in Fig. 4, in which the length of each horizontal road is 300 meters and 200 meters for each vertical road. Each road is designed to be bidirectional with two lanes. The traffic flow is generated as 150 vehicles in total with the help of SUMO *randomTrips.py*, and the vehicle load rate is one vehicle per second. At each intersection the traffic lights, which are controlled by SUMO default controller, are set in order to approach the real world.

Training stage The DRL agent is trained to find the minimum time cost route from the predefined initial road (red mark in Fig. 4) and the destination road (green mark in Fig. 4) in diversified traffic conditions by loading the DRL vehicle in different time steps. When there is no human driven vehicle in the network, the new DRL vehicle will not be loaded, and the next training episode will begin consecutively. The DRL vehicle will make a choice for the next road after going across the detector set on each roads when approaching each intersection.

Evaluation stage The well trained model, or the DRL vehicle router is applied to multiple DRL vehicles and guide them to make choices of next edge dynamically. In our experiment, we set 50% of the original 150 human vehicles to DRL vehicles. Additionally, because the time of leaving road and entering road for each vehicle is quite different to each other, the decision time is set to be equivalent for all the 75 DRL vehicles. By attenuating different time interval, we found that updating each DRL vehicle's next road decision every 30 simulation seconds has a relatively better performance, which is then used in the evaluation stage.

B. Compared methods

To evaluate our proposed DRL model, we compare three methods that are introduced as follows:

Dijkstra's Algorithm: Every 30 simulation seconds, we use Dijkstra's Algorithm to recompute the routes using the updated average travel speed on each road.

DQN: We applied DQN-based rerouting [3] to the same set of DRL vehicles as ours XRouting.

PPO: The basic PPO-based model to compare the advantages of the revised Transformer architecture used in our XRouting.

In terms of hyper-parameter settings, the number of attention head is set to 4 with the dim of each head is 32. The dim of MLP network is set to 100. The clip parameter ϵ is set to 0.3. The discount factor γ is set to 0.99. The learning rate is set to 0.0004, and the value loss coefficient is set to 0.00005. We train our model every 4096 batch size, and for each batch size, SGD iteration is designed to be conducted 4 times to update the network parameters. For the compared basic PPO algorithm, the policy/value network is comprised of two neural networks with RELU activation function, and the dim of the two networks are designed to 150 and 100 respectively. The rest parameters of the normal PPO algorithm, like learning rate, are set the same as our proposed model. All the three DRL algorithms are adjusted to be trained for 6500 episodes in total.

V. Experiments results and analysis

A. Algorithm Convergence

Fig 5 illustrates the reward convergence trends of the three compared DRL algorithms changing with the episodes. Each algorithm is trained for four epochs. Compared with PPO-based model, it is evidently shown that not only does the DQN-based model vibrate more, but it is also incapable of finding the optimal route for the DRL vehicle in this relatively more intractable traffic scenario. On the contrary, The PPO and XRouting are more stable. It is worth noting that the normal PPO shows a relatively lower convergent speed compared to our model. Specifically, our model reaches to -500 at about 2700 episodes in average, while the normal PPO-based model does not reach to this value until approximately 4000 episodes in average. With respect to the final reward value, our model has a slightly higher value compared to the normal PPO model, meaning that the normal-PPO find the subordinate optimal route for the DRL vehicle rather than the best one. In a

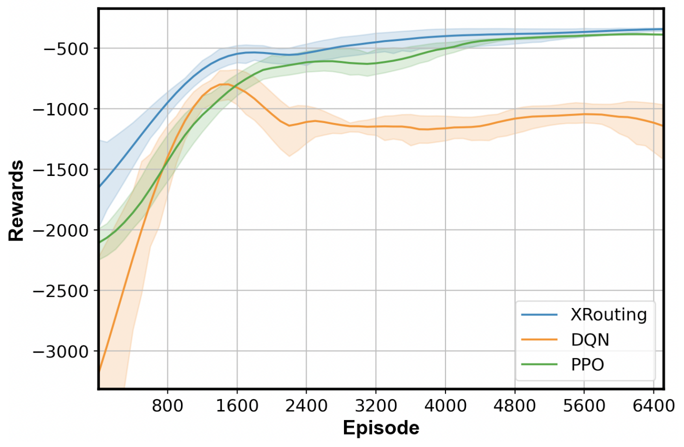

Fig. 5. The comparison of the reward convergence with the growing episodes among XRouting, DQN, and PPO. The shade represents the standard deviation.

nutshell, our model shows a better performance from either the stability or the final result perspective.

B. Traffic Impact

Table I shows that our model is capable of decreasing the average travel time of all vehicles by at most 4.09% compared to the traditional Dijkstra algorithm. For the average value of both CO2 emission and fuel consumption, compared with Dijkstra algorithm, our model saves by approximately 3.990%, which is nearly more than twice as many as the reduction resulted from normal PPO-based algorithm. Additionally, the average mile of route length is ameliorated by lowering 3.920%, which is fairly astounding. It is also noted that the DQN-based algorithm shows a quite bad performance when it applies to multiple vehicles rather than a single one in [3]. Specifically, all the average values of the four metrics increase by more than 7.000%, even deteriorating the original traffic conditions, which reflects the final convergent value of its reward in Fig. 5. In brief, it is proved that our proposed model outperforms in all the four metrics.

TABLE I
THE COMPARISON OF FOUR RE-ROUTING METHODS IN TERMS OF TRAVEL TIME, CO2 EMISSION, FUEL CONSUMPTION, AND ROUTE LENGTH. THE PERCENTAGE SHOWN IS COMPARED AGAINST DIJKSTRA.

	Travel Time (sec)	**CO2 (g/s)**	**Fuel (ml/s)**	**Route Length (m)**
Dijkstra	123.564	312.458	134.314	944.848
DQN	132.208 +7.000%	335.328 +7.320%	144.078 +7.270%	1019.478 +7.900%
PPO	120.015 -2.870%	306.982 -1.750%	132.040 -1.690%	929.107 -1.670%
XRouting	118.509 -4.090%	299.984 -3.990%	128.953 -3.990%	907.766 -3.920%

C. Explainability: which road feature matters?

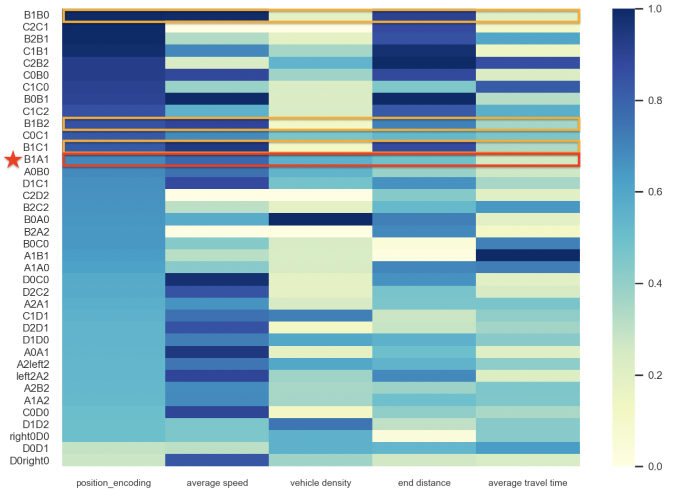

Fig. 6. The heat map of the normalized value of road attributes, when a DRL vehicle is at the road C1B1 while making its next road decision among the road B1B0, B1B2, B1C1, and B1A1 (suggested by XRouting). The x-axis is the name of road attribute, the y-axis is the road id.

To figure out which road attributes are predominant in a more intuitive way, the attributes for all roads of a DRL vehicle road decision trajectory are visualized in Fig. 6 as an example. Specifically, the DRL vehicle is on the road C1B1 currently, and the next road it chooses is B1A1 which is marked by the red star. The road attributes rows of other three choices for the DRL vehicle are marked by orange rectangles in Fig. 6. It is clear that B1A1 has the minimum value of the position encoding and end distance among the four available roads, meaning that crossing B1A1 can endow the DRL vehicle with the road that is geographically nearest to the destination, compared with the rest of three roads. Moreover, B1A1 also has a relatively less travel time value among the four roads. Nevertheless, it is interestingly found that B1A1 has the maximum value of vehicle density and minimum value of average speed, which seems that in this decision moment, the geographical elements have more dominated impact on the next road choice.

To gain more general insight from statistical perspective, we collect more than 2000 samples of input observation information and the corresponding output actions for "four choices cases" (i.e., when the DRL vehicle has to choose the next road from four possible ones. The similar conclusion can be found in "three-choice cases"). We rank the road attributes of all the candidate road selections for each decision according to the following rules: ordering the *average speed* attribute by numeric value from the largest to the smallest. Conversely, ordering the rest four attributes from the smallest to the largest. Finally, find the order value of the road that the DRL vehicle has chosen. For instance, in Fig. 6, for attribute *position encoding*, the ordered result of the road B1A1 is 1, while the order of attribute *average speed* is 4. Consequently, we calculate the main statistics value of all five attributes ranking for each decision and listed the results in Table II, while plotting the distribution in Fig. 7.

TABLE II
THE ORDER OF EACH INPUT ROAD ATTRIBUTE UNDER FOUR CANDIDATE NEXT ROAD CHOICES (2037 SAMPLES). THE HIGHER VALUE MEANS THE MORE IMPORTANCE THE ATTRIBUTE IS CONSIDERED FOR A RE-ROUTING DECISION. THE PERCENTAGE SHOWS THE PERCENTILE RANKING VALUE.

Attributes	**mean**	**25%**	**50%**	**75%**
Average Speed	2.345	1.000	2.000	3.000
Vehicle Density	2.347	1.000	2.000	3.000
Travelling Time	2.483	1.000	2.000	4.000
End Distance	2.550	2.000	3.000	4.000
Position Encoding	2.572	2.000	3.000	4.000

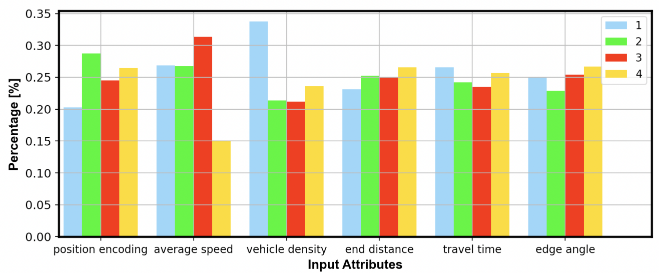

Fig. 7. The distribution of ranking values of all road attributes when re-routing among four choices.

According to Table II, we can see that the attribute *average speed* ranks the first in both two tales, meaning that it is the relatively most decisive attribute when it comes to the next road decision. The following attribute is *vehicle density*, *travelling time*, and *end distance*. It is worth noting that *position encoding* ranks the last. Because *position encoding* represented by dis_{veh_aim} as aforementioned, we can conclude that, in generic, the relative distance between the DRL vehicle and the destination by crossing a certain road has relatively minor impact on next road decision making process. While choose the next road that has the least vehicle occupancy often works better as shown in Fig. 7.

VI. Conclusions and Future Work

This paper proposes an explainable vehicle re-routing scheme for congestion avoidance called XRouting using deep reinforcement learning. The effectiveness of our approach has been demonstrated in a simulated urban scenario with quicker and more stable convergence when compared with the state-of-the-art DRL solution. We have also attempted to use explainable AI (XAI) to unbox the automated decision-making process of XRouting. Therefore, XRouting can guide human drivers to effectively reuse the re-routing knowledge learned from simulations to the realistic driving. Future work includes more extensive testing under various road conditions to improve the explainability to human drivers.

References

[1] S. Wang, S. Djahel, Z. Zhang, and J. McManis, "Next road rerouting: A multiagent system for mitigating unexpected urban traffic congestion," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 10, pp. 2888–2899, 2016.

[2] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, *et al.*, "Mastering the game of go with deep neural networks and tree search," *nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[3] S. Koh, B. Zhou, H. Fang, P. Yang, Z. Yang, Q. Yang, L. Guan, and Z. Ji, "Real-time deep reinforcement learning based vehicle navigation," *Applied Soft Computing*, vol. 96, p. 106694, 2020.

[4] Y. Geng, E. Liu, R. Wang, Y. Liu, W. Rao, S. Feng, Z. Dong, Z. Fu, and Y. Chen, "Deep reinforcement learning based dynamic route planning for minimizing travel time," in *2021 IEEE International Conference on Communications Workshops (ICC Workshops)*. IEEE, 2021, pp. 1–6.

[5] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[6] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[7] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[8] W. Kool, H. Van Hoof, and M. Welling, "Attention, learn to solve routing problems!" *arXiv preprint arXiv:1803.08475*, 2018.

[9] M. Nazari, A. Oroojlooy, L. Snyder, and M. Takác, "Reinforcement learning for solving the vehicle routing problem," *Advances in neural information processing systems*, vol. 31, 2018.

[10] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[11] E. Parisotto, F. Song, J. Rae, R. Pascanu, C. Gulcehre, S. Jayakumar, M. Jaderberg, R. L. Kaufman, A. Clark, S. Noury, *et al.*, "Stabilizing transformers for reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2020, pp. 7487–7498.

[12] J. Zhao, M. Mao, X. Zhao, and J. Zou, "A hybrid of deep reinforcement learning and local search for the vehicle routing problems," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 11, pp. 7208–7218, 2020.

[13] J. Pan, I. S. Popa, K. Zeitouni, and C. Borcea, "Proactive vehicular traffic rerouting for lower travel time," *IEEE Transactions on vehicular technology*, vol. 62, no. 8, pp. 3551–3568, 2013.

[14] S. Almagor and M. Lahijanian, "Explainable multi agent path finding," in *AAMAS*, 2020.