International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 7s
DOI: https://doi.org/10.17762/ijritcc.v11i7s.6988
***Article Received**: 28 March 2023 **Revised**: 16 May 2023 **Accepted**: 29 May 2023*

Improving the Performance of OLSR in Wireless Networks using Reinforcement Learning Algorithms

Seema Rani[1], Saurabh Charaya[2]
[1]Department of CSE
Om Streling Global University
Hisar, INDIA
seemacse202@osgu.ac.in
[2]Department of CSE
Om Streling Global University
Hisar, INDIA

Abstract— The Optimized Link State Routing Protocol is a popular proactive routing protocol used in wireless mesh networks. However, like many routing protocols, OLSR can suffer from inefficiencies and suboptimal performance in certain network conditions. To address these issues, researchers have proposed using reinforcement learning algorithms to improve the routing decisions made by OLSR. This paper explores the use of three RL algorithms - Q-Learning, SARSA, and DQN - to improve the performance of OLSR. Each algorithm is described in detail, and their application to OLSR is explained. In particular, the network is represented as a Markov decision process, where each node is a state, and each link between nodes is an action. The reward for taking an action is determined by the quality of the link, and the goal is to maximize the cumulative reward over a sequence of actions. Q-Learning is a simple and effective algorithm that estimates the value of each possible action in a given state. SARSA is a similar algorithm that takes into account the current policy when estimating the value of each action. DQN uses a neural network to approximate the Q-values of each action in a given state, providing more accurate estimates in complex network environments. Overall, all three RL algorithms can be used to improve the routing decisions made by OLSR. This paper provides a comprehensive overview of the application of RL algorithms to OLSR and highlights the potential benefits of using these algorithms to improve the performance of wireless networks.

Keywords- DQN, OLSR, Q-Learning, Reinforcement Learning, Routing, SARSA, Wireless.

I. INTRODUCTION

Reinforcement learning-based routing algorithm is a type of machine learning algorithm that uses trial-and-error learning to optimize routing decisions in computer networks. It is a type of unsupervised learning algorithm where the agent learns to make decisions based on its interaction with the environment. The reinforcement learning-based routing algorithm involves three primary components: the agent, the environment, and the reward function. The agent is responsible for making routing decisions based on the current state of the network. The agent interacts with the environment by taking actions and receiving feedback in the form of rewards or penalties. The environment represents the network topology and the current state of the network. The reward function provides feedback to the agent based on the quality of the routing decisions. The reinforcement learning-based routing algorithm works in an iterative process, where the agent learns from its experience and adjusts its routing decisions based on the feedback received from the environment. The agent seeks to maximize the cumulative reward over a period of time by learning the optimal routing policy. OLSR (Optimized Link State Routing) is a proactive routing protocol designed for mobile ad hoc networks (MANETs). It is a link-state routing protocol that uses a proactive approach to build and maintain a network topology map. OLSR is an improvement over traditional link-state routing protocols such as OSPF and IS-IS, which are designed for wired networks and are not well-suited for mobile ad hoc networks. OLSR builds a topology map of the network by exchanging topology information among nodes. Each node broadcasts its link state information, which includes the list of its neighbors and the quality of the links to those neighbors. The link quality is measured based on metrics such as signal strength, delay, and packet loss. OLSR uses a Multi-Point Relaying (MPR) technique to optimize the exchange of topology information. The MPRs are a subset of nodes that are responsible for relaying topology information to other nodes. By selecting a subset of nodes to act as MPRs, OLSR reduces the amount of overhead traffic in the network and improves the scalability of the protocol. Today, OLSR routing still has several utility in modern networks. OLSR is commonly used in ad-hoc and mesh networks due to its ability to quickly adapt to changes in network topology, such as the addition or removal of nodes, without requiring excessive bandwidth or computational resources. It is well-suited for emergency and disaster response networks, where communication infrastructure may be damaged or destroyed. OLSR can be used in IoT networks, where devices may be moving frequently and the network topology may change dynamically. It is also useful in rural and remote networks, where traditional infrastructure-based communication channels may be unavailable or unreliable. Overall, OLSR is a robust and efficient routing protocol for mobile ad hoc networks. The paper is organized as follows. In section 2, related work done in the area of OLSR is presented. Section 3 presents the proposed algorithm in which OLSR is modified using Q-Learning, SARSA and DQN based ML techniques. Section 4 provides results and analysis of proposed methods. Finally, in section 5 we draw conclusions and consider future work.

II. Related Work

J. Li and M. J. Neely [1] proposed a reinforcement learning-based algorithm for packet routing in communication networks. The proposed algorithm is shown to be effective in reducing network congestion and packet loss. J. Liu, et al. [2] proposed a reinforcement learning-based routing algorithm for Internet of Things networks. The proposed algorithm reduces energy consumption and improves network lifetime compared to traditional routing algorithms. S. Panwar and S. S. Ravi [3] presents a reinforcement learning-based approach to dynamic routing in telecommunication networks. The proposed algorithm outperforms traditional routing algorithms in terms of network utilization and packet delay. Zhang, et al. [4] proposed an efficient reinforcement learning-based routing algorithm for wireless sensor networks. The proposed algorithm reduces energy consumption and improves network lifetime.

Khan, et al. [5] proposed a Q-learning-based routing algorithm for wireless mesh networks. The proposed algorithm outperforms traditional routing algorithms in terms of packet delivery ratio and end-to-end delay. Zhang, et al. [6] proposed a novel reinforcement learning-based routing algorithm for vehicular ad hoc networks. The proposed algorithm outperforms traditional routing algorithms in terms of packet delivery ratio and end-to-end delay. Guo, et al. [7] proposed a deep reinforcement learning-based routing algorithm for software-defined networks. The proposed algorithm achieves high accuracy and performance compared to traditional routing algorithms. Li, et al. [8] proposes a deep reinforcement learning-based routing algorithm for cognitive radio networks. The proposed algorithm achieves high accuracy and performance compared to traditional routing algorithms. R. Liu, et al. [9] proposed an adaptive reinforcement learning-based routing algorithm for mobile ad hoc networks. The proposed algorithm adapts to network dynamics and achieves high performance in terms of packet delivery ratio and end-to-end delay. A. Kumar and V. Sharma [10] investigated the performance of OLSR routing protocol in mobile ad hoc networks (MANETs) and analyzed the impact of various network parameters on the OLSR routing protocol's performance, such as node density, mobility, and network size. The authors conclude that OLSR routing protocol performs well in large-scale MANETs with high node density, low mobility, and a significant number of intermediate nodes.

S. Goyal and R. Sharma's [11] proposed a fuzzy logic-based approach to enhance the performance of OLSR routing protocol in MANETs by introducing a new metric called link quality, which evaluates the link quality of each node in the network, and incorporate it into the OLSR routing protocol. The results show that the fuzzy logic-based OLSR routing protocol outperforms the standard protocol in terms of packet delivery ratio, end-to-end delay, and network throughput. Islam et al. [12] proposed a cross-layer framework for reliable communication using the OLSR routing protocol in wireless sensor networks (WSNs) by integrating the physical layer, data link layer, and network layer to enhance the OLSR routing protocol's reliability in WSNs. The results show that the cross-layer framework significantly improves the OLSR routing protocol's reliability in terms of packet delivery ratio and end-to-end delay. Li et al. [13] proposed a routing algorithm for mobile edge computing networks based on deep reinforcement learning. The proposed algorithm achieves better performance compared to traditional routing algorithms in terms of latency, throughput, and energy efficiency. Based on the review of literature on OLSR routing algorithms, it has been observed that existing algorithms have Limited scalability, Limited evaluation in real-world scenarios, lack of consideration for security issues and diverse network topologies which can be improved by implementing different types of Reinforcement algorithms.

III. Modelling of proposed RL based OLSR protocol

Reinforcement machine learning-based routing algorithms are effective in improving network performance in various contexts such as telecommunication networks, wireless sensor networks, wireless mesh networks, vehicular ad hoc networks, cognitive radio networks, and Internet of Things networks. Reinforcement learning-based routing protocols can improve the network lifetime and reduce energy consumption in wireless sensor networks. Reinforcement machine learning-based routing algorithms are effective and promising in improving network performance, and they can be further explored and developed in future research. Reinforcement learning can be implemented in OLSR to improve its routing decisions and network performance. One possible approach is to use reinforcement learning to optimize the selection of Multi-Point Relays (MPRs) in OLSR. In OLSR, the MPRs are responsible for relaying topology information to other nodes in the network. The selection of MPRs can significantly affect the overhead traffic and routing performance in the network. Reinforcement learning can be used to learn an optimal MPR selection policy that minimizes the overhead traffic and improves the network performance.

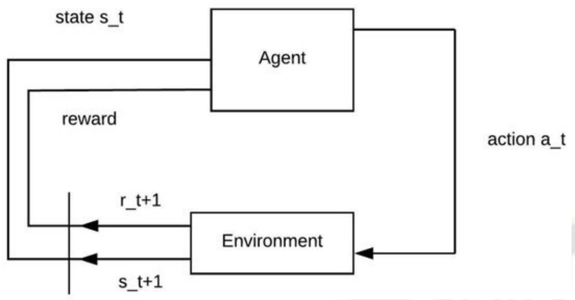

Figure 1.Reinforcement Learning Process

The generalization of the RL interaction process is shown in Figure 1, where the Agent interacts with the Environment through an a_t action. This interaction leads to a new s_t+1 state and generates a reward for the Agent. To implement reinforcement learning in OLSR, the following steps can be taken:

• *Define the state space*: The state space includes the current topology map and network conditions such as signal strength, delay, and packet loss.

• *Define the action space*: The action space includes the selection of MPRs based on the current state.

• *Define the reward function*: The reward function provides feedback to the reinforcement learning agent based on the quality of the selected MPRs. The reward can be based on metrics such as network utilization, packet delivery ratio, and energy consumption.

• *Train the reinforcement learning agent*: The reinforcement learning agent learns an optimal MPR selection policy by interacting with the environment and receiving feedback from the reward function. The agent can use techniques such as Q-learning, SARSA, or Deep Q-Networks to learn the optimal policy.

• *Update the MPR selection*: The reinforcement learning agent updates the selection of MPRs based on the learned policy, and the new topology information is broadcasted to other nodes in the network.

By implementing reinforcement learning in OLSR, the protocol can adapt to changing network conditions and optimize its routing decisions based on performance metrics. However, implementing reinforcement learning in OLSR requires significant computational resources, and the overhead traffic of the learning process should be carefully managed to avoid degrading network performance. Step-by-step proposed algorithm for implementing reinforcement learning in OLSR for optimizing MPR selection is as given below:

1. Define the state space:

- Let S be the state space that includes the current network topology map and network conditions such as signal strength, delay, and packet loss.

2.Define the action space:

- Let A be the action space that includes the selection of MPRs based on the current state.
- Let a_i be the i-th MPR selection action.

3.Define the reward function:

- Let R(s, a) be the reward function that provides feedback to the reinforcement learning agent based on the quality of the selected MPRs.
- The reward can be based on metrics such as network utilization, packet delivery ratio, and energy consumption.
- Let r_t be the reward received at time step t.

4.Initialize the Q-function:

- Let Q(s, a) be the Q-function that estimates the expected cumulative reward for taking action a in state s.
- Initialize Q(s, a) to a small random value.

5.Train the reinforcement learning agent:

For each time step t:

- Observe the current state s_t.
- Choose an action a_t based on the current Q-function using an exploration/exploitation trade-off.
- Execute the action a_t and observe the new state s_t+1 and the reward r_t+1.
- Update the Q-function using the Q-learning update rule:
- $Q(s_t, a_t) = Q(s_t, a_t) + \alpha[r_{t+1} + \gamma * \max(Q(s_{t+1}, a)) - Q(s_t, a_t)]$
- Update the current state s_t to s_t+1.

6.Update the MPR selection:

- Use the learned Q-function to select optimal MPRs based on current state.
- Broadcast the new topology information to other nodes in the network.

In the above algorithm, α is the learning rate, γ is the discount factor, and max(Q(s_t+1, a)) is the maximum expected cumulative reward for all actions a in the next state s_t+1. The

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 7s
DOI: https://doi.org/10.17762/ijritcc.v11i7s.6988
***Article Received**: 28 March 2023 **Revised**: 16 May 2023 **Accepted**: 29 May 2023*

exploration/exploitation trade-off determines whether to choose the optimal action or to explore new actions to gather more information about the environment. This algorithm can be implemented using various reinforcement learning techniques, such as Q-learning, SARSA, or Deep Q-Networks. The specific implementation details may vary depending on the chosen technique and the specific network context.

SARSA(State-Action-Reward-State-Action) is an on-policy algorithm that learns to estimate the optimal action-value function by iteratively updating its estimates using the observed rewards and transitions. SARSA selects actions based on its current estimates and continues to update them as it interacts with the environment. SARSA is often used in problems where the agent needs to interact with a stochastic environment, such as in robotics or game playing. Updation of Q-function using the SARSA update rule will be as :

$$Q(s\_t, a\_t) = Q(s\_t, a\_t) + \alpha[r\_t{+}1 + \gamma * Q(s\_t{+}1, a\_t{+}1) - Q(s\_t,a\_t)] \quad (1)$$

Now update the current state s_t to s_t+1 and the current action a_t to a_t+1. For updating the MPR selection, use the learned Q-function to select the optimal MPRs based on the current state and broadcast the new topology information to other nodes in the network. In the above algorithm, α is the learning rate, γ is the discount factor, and ε is the exploration rate that determines the balance between exploration and exploitation. The SARSA update rule uses the Q-value of the next state-action pair (s_t+1, a_t+1) to update the Q-value of the current state-action pair (s_t, a_t). The exploration policy uses the ε-greedy strategy, which chooses the optimal action with probability (1-ε) and a random action with probability ε.

Q-Learning, which is an off-policy algorithm and learns to estimate the optimal action-value function by choosing actions based on a different policy than the one being updated. Q-Learning updates its estimates using the maximum expected reward of the next state, regardless of the actual action taken. Q-Learning is often used in problems where the agent interacts with a deterministic environment, such as in navigation or control. Updation of the Q-function using the Q-learning update rule will be as:

$$Q(s\_t, a\_t) = Q(s\_t, a\_t) + \alpha[r\_t{+}1 + \gamma * \max(Q(s\_t{+}1, a)) - Q(s\_t, a\_t)] \quad (2)$$

Now update the current state s_t to s_t+1. Further for updating the MPR selection, Use the learned Q-function to select the optimal MPRs based on the current state and Broadcast the new topology information to other nodes in the network. In this algorithm, α is the learning rate, γ is the discount factor, max(Q(s_t+1, a)) is the maximum expected cumulative reward for all actions a in the next state s_t+1, and epsilon is the exploration probability. The exploration/exploitation trade-off determines whether to choose the optimal action or to explore new actions to gather more information about the environment.

DQN (Deep Q-Network) is an extension of Q-Learning that uses a deep neural network to approximate the action-value function. DQN has been shown to achieve state-of-the-art results in a variety of complex RL problems, such as playing Atari games or navigating complex mazes. DQN uses experience replay and target networks to stabilize the learning process and avoid overfitting. Here the target Q-values for the sampled transitions using the DQN target network is computer as :

$$y\_j = r\_j{+}1 + \gamma * \max\_a Q\_target(s\_j{+}1, a) \quad (3)$$

For Computing the predicted Q-values for the sampled transitions using the DQN:

$$Q\_pred(s\_j, a\_j) = DQN(s\_j)[a\_j] \quad (4)$$

DQN weights are updated using the mean squared error loss between the target Q-values and the predicted Q-values:

$$L = 1/N * sum\_i (y\_i - Q\_pred(s\_i, a\_i))^\wedge 2 \quad (5)$$

Now for updating the MPR selection, use the learned DQN to select the optimal MPRs based on the current state and broadcast the new topology information to other nodes in the network. In the above algorithm, the DQN target network is a separate copy of the DQN that is periodically updated with the DQN weights. This helps to stabilize the training process by reducing the correlations between the target Q-values and the predicted Q-values**.**

IV. RESULTS & PERFORMANCE ANALYSIS OF PROPOSED ALGORITHM

Performance of proposed method is obtained by using Network Simulator NS-2. The simulations are performed for Q-Learning, SARSA and DQN based modified OLSR routing protocol. The protocol available with network simulator 3.22 is modified by implementing technique proposed in Section 3 and performance is analyzed for various parameters like End-to-End Delay, Throughput and PDR. In this scenario an area of 250x250 m2 has been used. Simulation is performed for duration of 10 seconds by giving a pause of 2 seconds for Random Walk Model. Channel capacity is 1Mbps and size of packet is 1024 bytes. A total of 25 nodes are allowed in this simulation with varying node density i.e. nodes are varies from 5 to 25. These nodes move at a speed of 20m/sec. 802.11n is used as the MAC protocol.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 7s
DOI: https://doi.org/10.17762/ijritcc.v11i7s.6988
***Article Received**: 28 March 2023 **Revised**: 16 May 2023 **Accepted**: 29 May 2023*

TABLE I. SIMULATION PARAMETERS

Parameters	**Value**	**Parameters**	**Value**
Routing Protocol	OLSR	Packet size	1024 Byte
Area	250 x 250	Data rate	1Mbps
Nodes	5,10,15,20,25	Mobility Models	Random Walk
MAC Protocol	802.11 n	Pause time	2 sec.
Simulation Time	10 sec.	Node speed	20m/sec.

Performance of modified OLSR is evaluated for Q-Learning, SARSA and SARSA based ML techniques at varying number of active nodes from 5 to 25.

A. *Packet Delivery Ratio (PDR %)*

Table 2 and corresponding graph in figure 2 shows the evaluated values of PDR in OLSR protocol when number of active nodes are varied from 5 to 25 for standard OLSR protocol and modified Q-Learning, SARSA and DQN based OLSR protocol for Random Walk mobility model. All three RL based algorithm perform better then normal OLSR protocol and among three RL based techniques, DQN performs best in terms of Packet Delivery Ratio. As number of active nodes increases, percentage of PDR decreases.

TABLE II. VALUES OF PDR IN MODIFIED OLSR PROTOCOL

PDR in %age				
No. of Active nodes	***OLSR***	***Q-Learning based OLRS***	***SARSA based OLSR***	***DQN based OLSR***
5	85	87.5	90.1	93.5
10	79	84.5	87.8	90.1
15	76	78	81.3	85
20	73	77	83	84
25	71.5	75	79	83

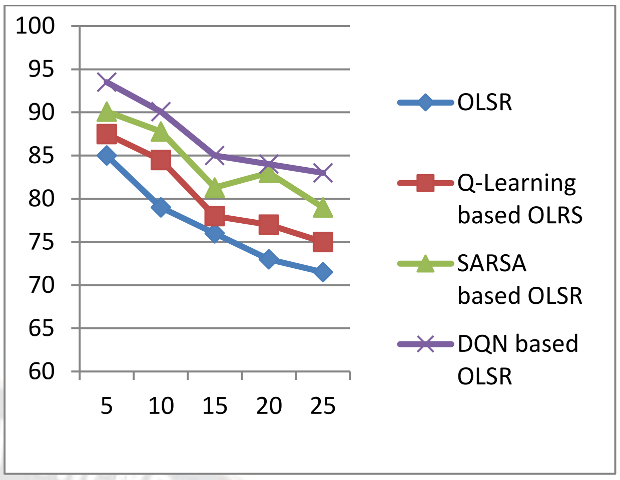

Figure 2 Graph for PDR in Modified OLSR protocol (PDR in %age versus Active users)

B. *Throughput*

Table 3 and corresponding graph in figure 3 shows the evaluated values of throughput in OLSR protocol when number of active nodes are varied from 5 to 25 for standard OLSR protocol and modified Q-Learning, SARSA and DQN based OLSR protocol for Random Walk mobility model. All three RL based algorithm perform better then normal OLSR protocol and among three RL based techniques, DQN performs best in terms of throughput. As number of active nodes increases throughput increases.

TABLE III. VALUES OF THROUGHPUT IN MODIFIED OLSR PROTOCOL

Throughput in kbps				
No. of Active nodes	**OLSR**	**Q-Learning based OLSR**	**SARSA based OLSR**	**DQN based OLSR**
5	4800	5250	5800	6390
10	6500	8230	8900	9400
15	10200	11650	12780	13890
20	12900	14100	15890	16890
25	16100	17500	18993	19780

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 7s
DOI: https://doi.org/10.17762/ijritcc.v11i7s.6988
***Article Received**: 28 March 2023 **Revised**: 16 May 2023 **Accepted**: 29 May 2023*

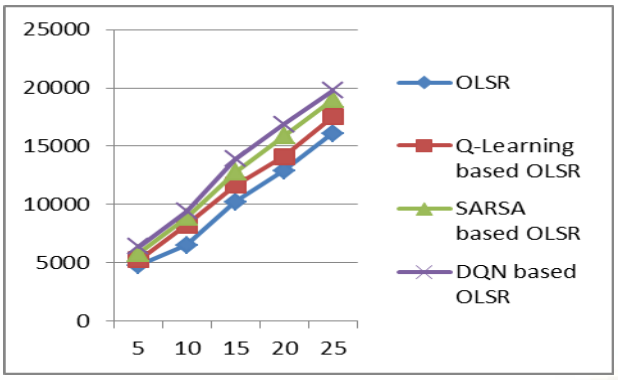

Fig 3. Graph for Throughput in modified OLSR protocol (throughput in kbps versus Active users)

C. *End-to-End* *Delay*

Table 4 and corresponding graph in figure 4 shows the evaluated values of delay in OLSR protocol when number of active nodes are varied from 5 to 25 for standard OLSR protocol and modified Q-Learning, SARSA and DQN based OLSR protocol for Random Walk mobility model. All three RL based algorithm perform better then normal OLSR protocol and among three RL based techniques, Q-Learning performs best in terms of delay. As number of active nodes increases delay also increases.

TABLE IV. VALUES OF END TO END DELAY IN MODIFIED OLSR PROTOCOL

Delay in Sec				
No. of Active nodes	**OLSR**	**Q-Learning based OLSR**	**SARSA based OLSR**	**DQN based OLSR**
5	0.041	0.035	0.038	0.039
10	0.055	0.05	0.051	0.054
15	0.065	0.06	0.063	0.064
20	0.07	0.065	0.068	0.069
25	0.081	0.073	0.076	0.079

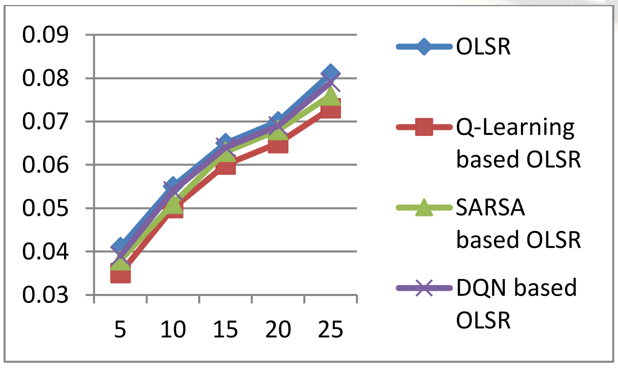

Figure 4 Graph for End to End delay in modified OLSR protocol (Delay in Sec versus Active users)

From the results it is found that DQN based OLSR performs best in terms of PDR and Throughput but lags in terms of delay. Q-Learning based OLSR performs better then SARSA algorithm. In summary, SARSA is an on-policy algorithm that learns to estimate the optimal action-value function, Q-Learning is an off-policy algorithm that learns to estimate the optimal action-value function, and DQN is an extension of Q-Learning that uses a deep neural network to approximate the action-value function. Q-learning and SARSA are simpler and more computationally efficient than DQN, but they may struggle with larger and more complex state spaces. Q-learning is often used for problems with discrete action spaces, while SARSA is better suited for continuous action spaces. Each algorithm has its own strengths and weaknesses and is suited for different types of RL problems.

V. CONCLUSION

This paper has discussed the application of reinforcement learning (RL) algorithms - Q-Learning, SARSA, and DQN - to improve the performance of the Optimized Link State Routing Protocol (OLSR) in wireless networks. By representing the network as a Markov decision process and using the quality of the link as the reward, these algorithms can learn the optimal path between two nodes based on the current network conditions. While all three algorithms can be effective in improving the performance of OLSR, the specific algorithm to use will depend on the characteristics of the network and the performance metrics that are most important to optimize. For example, Q-Learning may be more appropriate in less complex network environments, while DQN may be better suited for more complex networks. As far as PDR and throughput is concerned, DQN performs better then other two while Q-Learning performs best in terms of delay. Overall, the application of RL algorithms to OLSR provides a promising approach to improve the routing decisions made by OLSR, leading to more efficient and effective wireless mesh networks. Further research in this area could explore the use of other RL algorithms or combinations of algorithms to further enhance the performance of OLSR and other routing protocols.

REFERENCES

[1] J. Li and M. J. Neely, “Reinforcement Learning for Packet Routing in Communication Networks”, IEEE Transactions on Neural Networks and Learning Systems, Vol. 28, No. 11, pp. 2650-2662 2017.

[2] Prof. Amruta Bijwar. (2016). Design and Analysis of High Speed Low Power Hybrid Adder Using Transmission Gates. International Journal of New Practices in Management and Engineering, 5(03), 07 - 12. Retrieved from http://ijnpme.org/index.php/IJNPME/article/view/46

[3] J. Liu, Y. Wang, X. Gao, and G. Hu, “A Reinforcement Learning-Based Routing Algorithm for Internet of Things Networks”, IEEE Access, Vol. 6, 2018, pp. 14039-14051, 2018.

[4] S. Panwar and S. S. Ravi, “A Reinforcement Learning Approach to Dynamic Routing in Telecommunication Networks”, IEEE Transactions on Neural Networks and Learning Systems, Vol. 30, Issue 6, pp. 1794-1808, 2019.

[5] Ólafur, S., Nieminen, J., Bakker, J., Mayer, M., & Schmid, P. Enhancing Engineering Project Management through Machine Learning Techniques. Kuwait Journal of Machine Learning, 1(1). Retrieved from http://kuwaitjournals.com/index.php/kjml/article/view/112

[6] H. Zhang, Y. Wang, H. Liu, and Z. Chen, “An Efficient Reinforcement Learning-Based Routing Algorithm for Wireless Sensor Networks”, IEEE Access, Vol. 7, pp. 27335-27347, 2019.

[7] M. A. Khan, S. A. Madani, S. A. Hussain, and M. S. Alresheedi, “A Q-Learning-Based Routing Algorithm for Wireless Mesh Networks”, IEEE Access, Vol. 7, pp. 139387-139401, 2019.

[8] X. Zhang, Y. Wang, X. Zhang, and J. Li, “A Novel Reinforcement Learning-Based Routing Algorithm for Vehicular Ad Hoc Networks”, IEEE Transactions on Vehicular Technology, Vol. 68, Issue 11, pp. 10619-10633, 2019.

[9] Y. Guo, S. Wang, J. Zhang, and Y. Zhang, “A Deep Reinforcement Learning-Based Routing Algorithm for Software-Defined Networks”, IEEE Access, Vol. 8, pp. 2380-2392, 2020.

[10] Kshirsagar, D. R. . (2021). Malicious Node Detection in Adhoc Wireless Sensor Networks Using Secure Trust Protocol. Research Journal of Computer Systems and Engineering, 2(2), 12:16. Retrieved from https://technicaljournals.org/RJCSE/index.php/journal/article/view/26

[11] Kandula, A. R. ., Sathya, R. ., & Narayana, S. . (2023). Multivariate Analysis on Personalized Cancer Data using a Hybrid Classification Model using Voting Classifier. International Journal of Intelligent Systems and Applications in Engineering, 11(1), 354–362. Retrieved from https://ijisae.org/index.php/IJISAE/article/view/2546

[12] Q. Li, Y. Li, Z. Liang, and W. Wang, “A Deep Reinforcement Learning-Based Routing Algorithm for Cognitive Radio Networks”, IEEE Transactions on Cognitive Communications and Networking, Vol. 6, Issue 3, pp. 936-950, 2020.

[13] Mei Chen, Machine Learning for Energy Optimization in Smart Grids , Machine Learning Applications Conference Proceedings, Vol 2 2022.

[14] R. Liu, Y. Zhang, X. Wang, and Q. Wang, “An Adaptive Reinforcement Learning-Based Routing Algorithm for Mobile Ad Hoc Networks”, IEEE Access, Vol. 8, pp. 158409-158421, 2020.

[15] A. Kumar and V. Sharma, "Performance Analysis of OLSR Routing Protocol in Mobile Ad Hoc Networks," Proc. of IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS), pp. 1-6, 2022.

[16] S. Goyal and R. Sharma, "Enhancing the Performance of OLSR Routing Protocol Using Fuzzy Logic in Mobile Ad Hoc Networks," Proc. of IEEE International Conference on Computing, Electronics & Communications Engineering (IEEE ICCECE), pp. 1-5. 2022.

[17] M. K. Islam, K. S. Kwak, and S. S. Kim, "A Cross-Layer Framework for Reliable Communication Using OLSR Routing Protocol in Wireless Sensor Networks," Proc. of IEEE International Conference on Consumer Electronics (ICCE), pp. 1-5, 2022.

[18] W. Li, Y. Zhang, C. C. Han, and H. Zheng, “Deep Reinforcement Learning-Based Routing for Mobile Edge Computing Networks”, IEEE Transactions on Mobile Computing, Vol. 21, Issue 3, pp. 1792-1805, 2022.