



DQN-CAMR: A Deep Q-Network Based Congestion-Aware Multipath Routing Framework for Quality-of-Service Provisioning in Mobile Ad-Hoc Networks

¹P. Kalyanchakravarthi, ²P. Surya, ³Dr. Thappeta Praveen Kumar Reddy, ⁴Bhavya K

¹Asst.prof., Dept of ECE, GMRI Deemed to be University, Andhra Pradesh-532127. E-mail: kalyanecebujji@gmail.com

²Associate Professor, ECE, RISE Krishna Sai Prakasam Group of Institutions, Andhra Pradesh-523272.

E-mail: suryame3020@gmail.com

³Assistant Professor, Department of AIDS, T.J.S. Engineering College, Tamil Nadu 601206. E-mail: preddy@gmail.com

⁴Department of Mathematics, NMAM INSTITUTE OF TECHNOLOGY, NITTE (Deemed to be University), 574110, Karnataka, India. E-mail: bhavyak@nitte.edu.in

Abstract

Reactive single-path protocols such as AODV commit an entire flow to one route for its lifetime, so when node mobility or bursty traffic pushes queue occupancy at an intermediate node past its buffer capacity, every flow sharing that node suffers correlated delay growth and tail packet loss until a new route is discovered. Multipath extensions such as AOMDV reduce this effect by maintaining backup routes, but typically split traffic using static or hop-count-based ratios that do not react to real-time queue state, leaving the underlying congestion-avoidance problem largely unsolved. This paper formulates per-flow traffic distribution across multiple discovered paths as a constrained Markov Decision Process whose objective is to minimize a weighted combination of end-to-end delay, packet loss, and maximum link utilization, and proposes DQN-CAMR, a Deep Q-Network based Congestion-Aware Multipath Routing framework that learns, from local queue-occupancy, link-quality, and hop-count observations, how to split each flow's traffic across up to three concurrently maintained paths. Congestion is estimated directly from MAC-layer queue occupancy rather than inferred indirectly from delay, allowing the agent to react before queue overflow causes loss rather than after. DQN-CAMR extends AOMDV route-discovery with a lightweight congestion-metric field and replaces its static split ratio with a compact feed-forward Q-network re-evaluated every route-maintenance interval. Simulated in NS-3 across 50-150 nodes and 2-10 Mbps offered load against AOMDV, AODV, and an MDQ-inspired congestion-aware baseline adapted from SDN literature, DQN-CAMR sustains 15-30% higher aggregate throughput at high load, cuts average end-to-end delay by 18-32% at 20-25 concurrent flows, reduces packet loss ratio by roughly half relative to AOMDV at 10 Mbps offered load, and maintains a Jain's fairness index above 0.92 as flow count grows to 25, indicating that congestion-aware, learning-based traffic splitting is a practical route to QoS-aware routing in mobile ad-hoc networks without requiring centralized control.

Keywords—MANET, multipath routing, deep reinforcement learning, deep Q-network, congestion control, AOMDV, quality of service, load balancing

1. Introduction

Mobile Ad-hoc Networks (MANETs) form multi-hop paths through cooperating mobile nodes without fixed infrastructure, which makes route capacity inherently limited by the buffering and forwarding capability of whichever nodes happen to lie on the active path. When several flows converge on a common relay, or when node mobility temporarily concentrates traffic onto a smaller set of surviving links, the queue at that relay can fill faster than it drains, producing a burst of tail packet drops and a sharp rise in delay that affects every flow passing through the congested node, not only the flow that triggered it. Reactive single-path protocols such as the Ad-hoc On-Demand Distance Vector (AODV) protocol have no mechanism to detect this condition before it produces loss, and by the time a route error propagates back to the source, the flow has already experienced the congestion event it is trying to avoid.

Multipath protocols such as AOMDV address part of this problem by discovering and caching multiple loop-free, mostly disjoint paths to a destination during a single route-discovery flood, so that a source has an alternative available immediately after a route error rather than waiting for a fresh flood. However, the traffic-splitting rule most multipath protocols use — equal splitting, or splitting weighted by hop count or a fixed reliability estimate computed once at discovery time — does not observe or react to how full each path's intermediate queues currently are. A path can therefore continue to receive its assigned share of traffic well after its queues have begun to build up, simply because the split ratio was fixed before the congestion event began.

Deep reinforcement learning (DRL) has shown, in software-defined and satellite networking contexts, that a learned policy conditioned on real-time queue and link-quality observations can redistribute traffic

across paths more responsively than a static split rule, consistently improving throughput and reducing loss under bursty load. Translating this idea to MANETs is not immediate, however, because centralized DRL controllers common in SDN settings assume a global view of network state that a distributed, infrastructure-less MANET does not provide, and because the neural architectures used in several prior DRL routing studies are too heavy to re-evaluate at every route-maintenance interval on a battery-constrained mobile node.

This paper addresses the specific problem of per-flow traffic splitting across a small set of concurrently maintained multipath routes under time-varying queue occupancy, using only observations available locally at the source node. The contributions of this paper are:

- 1) A congestion-aware extension to AOMDV route discovery in which RREQ/RREP packets carry a queue-occupancy field, giving the source a per-path congestion estimate before any data is transmitted.
- 2) A compact Deep Q-Network, sized to run within the control-plane budget of a mobile node, that maps discretized queue-occupancy, link-quality, and hop-count observations for up to three candidate paths to a traffic-split action, re-evaluated every route-maintenance interval.
- 3) A reward function that combines normalized delay, loss, and maximum path-link-utilization terms so that the learned policy explicitly avoids concentrating traffic on any single congesting path rather than only reacting after loss is observed.

The remainder of this paper is organized as follows. Section II reviews recent congestion-aware and DRL-based multipath routing work. Section III formulates the traffic-splitting problem. Section IV presents the DQN-CAMR framework and its algorithm. Section V analyzes its computational complexity. Section VI describes the simulation environment. Section VII reports results. Section VIII analyzes robustness and scalability, and Section IX concludes.

II. RELATED WORK

Congestion-adaptive routing for MANETs has recently been examined by Rathod et al. [1], whose congestion-adaptive routing protocol dynamically adjusts routes based on real-time congestion state and reports substantial end-to-end delay and delivery-ratio improvements over DSR, TORA, OLSR, and AODV, though the underlying mechanism remains a single-path route switch rather than concurrent multipath traffic splitting. Arun and Jayanthi [2] proposed an adaptive congestion- and energy-aware multipath routing scheme built on stable link prediction, reporting higher throughput and lower delay through path selection that anticipates link breakage before it occurs, which motivates the queue- and link-quality-based state representation adopted in this paper.

Learning-based congestion control has been explored more extensively in SDN and wireless sensor network settings than in MANETs specifically. Zala et al. [3] proposed a deep-reinforcement-learning-based congestion control algorithm for wireless networks, and Tran et al. [4] developed a DRL-based QoS routing protocol exploiting cross-layer design for cognitive radio MANETs, reporting that cross-layer state observation (combining physical, MAC, and network-layer signals) improves routing decisions over network-layer-only state. Safari et al. [5] proposed a cross-layer adaptive fuzzy AODV variant, illustrating an alternative, non-learning approach to the same cross-layer sensing problem. Choi et al. [6] applied DRL to minimize end-to-end delay through distributed routing in MANETs, and Keum and Ko [7] combined trust estimation with Q-learning for mission-critical wireless sensor network routing, showing that a tabular or shallow Q-learning formulation can be effective when the state space is kept small enough for a resource-constrained device.

Multi-agent DRL routing has also been studied for highly dynamic tactical and satellite networks: Kaviani et al. [8] proposed DeepCQ+, a multi-agent DRL routing scheme for highly dynamic networks, and a QoS-congestion-aware DRL approach for SDN multipath routing (MDQ) [9] integrates a multi-objective loss function combining link and queue metrics to select paths and allocate traffic proportions, which we adapt as a representative congestion-aware baseline in Section VI since its queue-aware objective is conceptually the closest prior work to the state representation used in this paper. More broadly, DRL-based multipath routing has been applied to LEO megaconstellation networks [13], where time-varying topology motivates a similar move away from static split ratios toward learned, state-conditioned traffic allocation.

Across this literature, two gaps are consistently present for the MANET setting specifically: first, most DRL routing schemes assume either a centralized controller with a global topology view (reasonable in SDN, not available in a MANET) or a neural architecture too large to re-evaluate at every route-maintenance interval on a mobile node; and second, congestion is frequently inferred indirectly from observed delay rather than measured directly from queue occupancy, which delays reaction until after a queue has already begun to build. DQN-CAMR is designed specifically around these two constraints: a fully distributed, per-source decision using only locally observable queue-occupancy and link-quality signals, and a network small enough to execute within a MANET node's control-plane budget.

III. Problem Formulation

A. System Model

We consider a MANET of N mobile nodes moving under the Random Waypoint model. For an active source-destination pair (s, d) , AOMDV route discovery returns a set of up to K disjoint or partially disjoint paths

$P_{s,d} = \{p_1, \dots, p_K\}$. Each intermediate node n_i maintains a MAC-layer output queue of capacity $Q_{\max}$ and reports its instantaneous occupancy $q_i(t) \in [0, Q_{\max}]$ to upstream neighbors via an extended HELLO/RREP field. Each path p_j is therefore characterized, at decision time t , by a congestion level $C_j(t) = \max_{i \in p_j} q_i(t)/Q_{\max}$ (the most congested hop dominates the path), a link-quality estimate $L_j(t)$ based on recent packet delivery ratio on the path's links, and a hop count H_j .

B. Formal Problem Statement

Let $x_j(t) \in [0, 1]$ denote the fraction of the source's offered traffic assigned to path p_j at time t , with $\sum_j x_j(t) = 1$. The traffic-splitting problem solved at every route-maintenance interval is to choose the split vector $x(t) = (x_1(t), \dots, x_K(t))$ that minimizes a weighted combination of expected delay, expected loss, and worst-case link utilization across the chosen paths:

$$x^*(t) = \operatorname{argmin}_x x(t) \sum_j x_j(t) \cdot [\beta_1 \cdot D_j(t) + \beta_2 \cdot \text{Loss}_j(t)] + \beta_3 \cdot \max_j C_j(t)$$

subject to $\sum_j x_j(t) = 1$, $x_j(t) \geq 0$, $C_j(t) \leq C_{\max} \quad \forall p_j \in P_{s,d}$

where $D_j(t)$ is the current estimated delay on path p_j , $\text{Loss}_j(t)$ its recent loss rate, $C_{\max}$ a congestion ceiling above which a path is excluded from the split entirely, and $\beta_1, \beta_2, \beta_3$ scalarization weights. Because $D_j(t)$, $\text{Loss}_j(t)$, and $C_j(t)$ are all functions of traffic that has not yet been sent — the act of assigning $x_j(t)$ itself changes future queue occupancy on p_j — this is a closed-loop control problem rather than a one-shot optimization, and the optimal split at time t depends on the sequence of past splits through their effect on current queue state. This coupling between action and future state is precisely the structure a Markov Decision Process captures, and Section IV reformulates the problem accordingly rather than attempting to solve the static program above independently at each interval.

IV. PROPOSED DQN-CAMR FRAMEWORK

A. State, Action, and Reward Design

The MDP state s_t at a source node is the concatenation of, for each of up to $K = 3$ candidate paths, a discretized congestion level (5 bins), a discretized link-quality estimate (4 bins), and normalized hop count, giving a fixed-length state vector regardless of how many paths are currently active (unused path slots are zero-filled and masked in the action selection). The action a_t is selected from a discrete set of pre-defined split templates (e.g., $\{100/0/0\}$, $\{70/20/10\}$, $\{50/30/20\}$, $\{34/33/33\}$, and permutations thereof) rather than a continuous split, which keeps the action space small enough for a standard DQN output layer and avoids the training instability associated with continuous action spaces on a resource-constrained learning agent.

The reward is computed one route-maintenance interval after an action is applied, once the resulting delay, loss, and queue occupancy are observable:

$$r_t = -\beta_1 \cdot \bar{D}(t) - \beta_2 \cdot \bar{\text{Loss}}(t) - \beta_3 \cdot \max_j C_j(t) + \beta_4 \cdot \text{Thr}(t)$$

where $\bar{D}(t)$ and $\bar{\text{Loss}}(t)$ are the traffic-weighted average delay and loss across the split, $\max_j C_j(t)$ penalizes concentrating traffic on the most congested available path even if that path currently has acceptable delay, and $\text{Thr}(t)$ is the achieved throughput, included so that the agent is not rewarded for trivially avoiding congestion by under-utilizing the network.

B. Network Architecture and Training

The Q-network is a feed-forward network with two hidden layers of 64 and 32 ReLU units, taking the state vector described above as input and producing one Q-value per split-template action. This architecture was deliberately kept small relative to convolutional or graph-based DRL routing architectures used in SDN and satellite settings [3], [9], [13], since a MANET source node must be able to run inference every route-maintenance interval (typically 1-2 s) without materially affecting battery life. Training uses experience replay and a target network updated every 500 steps, following standard DQN practice [10], with the source node logging (s_t, a_t, r_t, s_{t+1}) tuples locally and performing periodic offline updates rather than online gradient steps on every interval, so that training cost is decoupled from the real-time routing decision.

C. Congestion-Aware Route Discovery and Maintenance

RREQ and RREP packets are extended with a cumulative congestion field, updated as the maximum queue occupancy ratio observed at any node along the path so far, in addition to the hop-count and sequence-number fields AOMDV already maintains. A path whose cumulative congestion exceeds $C_{\max}$ at discovery time is still cached (in case all other candidates later become congested) but is excluded from the initial split until its congestion estimate falls back below threshold. During data transmission, each source node re-evaluates $C_j(t)$ for its active paths every route-maintenance interval using freshly received HELLO updates, feeds the resulting state into the Q-network, and applies the resulting split to the next interval's traffic, closing the sense-decide-act loop described in Section III-B.

D. Per-Interval Decision Procedure

Algorithm 1 summarizes the decision executed by a source node at every route-maintenance interval.

Algorithm 1: DQN-CAMR Traffic-Split Selection

Input: candidate paths $P_{s,d}$, Q-network $Q(\cdot; \theta)$
Output: traffic-split action a^*

```

1: for each  $p_j \in \mathcal{P}_s, d$  do
2:    $C_j(t) \leftarrow \max$  queue occupancy along  $p_j \triangleright$  Sec. IV-C
3:    $L_j(t) \leftarrow$  recent link-quality estimate
4:   if  $C_j(t) > C_{\max}$  then exclude  $p_j$  from split
5: end for
6:  $st \leftarrow [\text{discretize}(C_j, L_j, H_j)]_{j=1..K} \triangleright$  Sec. IV-A
7: with probability  $\varepsilon$ :  $a^* \leftarrow$  random valid split
8: else:  $a^* \leftarrow \arg\max_a Q(st, a; \theta)$ 
9: apply split  $a^*$  to next interval's traffic
10: observe  $\bar{D}(t)$ ,  $\text{Loss}(t)$ ,  $\text{Thr}(t)$  after interval elapses
11:  $rt \leftarrow -\beta_1 \cdot \bar{D}(t) - \beta_2 \cdot \text{Loss}(t) - \beta_3 \cdot \max_j C_j(t) + \beta_4 \cdot \text{Thr}(t)$ 
12: store  $(st, a^*, rt, st+1)$  in replay buffer
13: periodically sample minibatch and update  $\theta$ 
14: return  $a^*$ 

```

V. COMPLEXITY ANALYSIS

Each decision requires $O(K)$ congestion and link-quality updates (Algorithm 1, lines 1-5) followed by a single forward pass through the Q-network. With two hidden layers of widths $h_1 = 64$ and $h_2 = 32$, input dimension $d_{in} = 3K$ (three features per candidate path, $K \leq 3$), and $|A|$ split-template actions, the forward-pass cost is $O(d_{in} \cdot h_1 + h_1 \cdot h_2 + h_2 \cdot |A|)$, which for the parameters used in evaluation ($d_{in} = 9$, $h_1 = 64$, $h_2 = 32$, $|A| = 8$) amounts to fewer than 3,000 multiply-accumulate operations per decision — several orders of magnitude below the cost of a single AODV route-discovery flood and well within the inference budget of a low-power mobile processor. Memory cost is dominated by the network weights, $O(d_{in} \cdot h_1 + h_1 \cdot h_2 + h_2 \cdot |A|) \approx 3,000$ parameters, independent of network size N , in contrast to graph-based DRL routing architectures whose parameter count typically scales with the number of nodes or edges represented in the input graph. Training cost (replay buffer sampling and backpropagation) is incurred only during the periodic offline update in line 13 and does not affect the real-time decision latency analyzed above.

VI. SIMULATION ENVIRONMENT

DQN-CAMR was implemented as an extension to AODMV in NS-3 and compared against three baselines: unmodified AODV [12] (single-path, reactive), unmodified AODMV [2] (multipath, static hop-count-weighted split), and an MDQ-inspired congestion-aware baseline [9] adapted from its original SDN setting by replacing centralized controller state with the same locally observable queue and link-quality fields used by DQN-CAMR, so that the comparison isolates the effect of the learned policy rather than the effect of additional state information. Table I summarizes the simulation parameters.

Parameter	Value
Simulator	NS-3.40
Simulation area	1000 m $\times$ 1000 m
Number of nodes	50, 100, 150
Mobility model	Random Waypoint, 0-15 m/s
MAC / PHY	IEEE 802.11 DCF
Queue capacity ($Q_{\max}$)	50 packets
Max candidate paths (K)	3
Offered load	2-10 Mbps
Concurrent flows	5-25
Simulation time	600 s per run
Hidden layers	64, 32 (ReLU)
Learning rate	1e-3 (Adam)
Discount factor (γ)	0.95
Congestion ceiling ($C_{\max}$)	0.85

TABLE I. Simulation Parameters

Five metrics were used for evaluation: aggregate throughput, average end-to-end delay, packet loss ratio, jitter, and Jain's fairness index computed across concurrent flows' achieved throughput, the last capturing whether the scheme avoids starving some flows while over-serving others under load. Each configuration was averaged over 20 independent runs with different random seeds.

VII. Results and Discussion

A. Aggregate Throughput

Fig. 1 plots aggregate throughput against offered load at 100 nodes. DQN-CAMR tracks offered load closely up to about 6 Mbps and then saturates around 7.6 Mbps at 10 Mbps offered load, 15-30% above AODV and 8-13% above AOMDV at the same load, because its split decisions redirect traffic away from paths as their queues approach C_{max} rather than continuing to push traffic onto an already-filling queue until loss occurs. The MDQ-adapted baseline tracks DQN-CAMR closely at low load but falls behind by 8-13% at 8-10 Mbps, consistent with its split-ratio objective being tuned for the more stable topology assumptions of its original SDN setting rather than MANET-scale mobility.

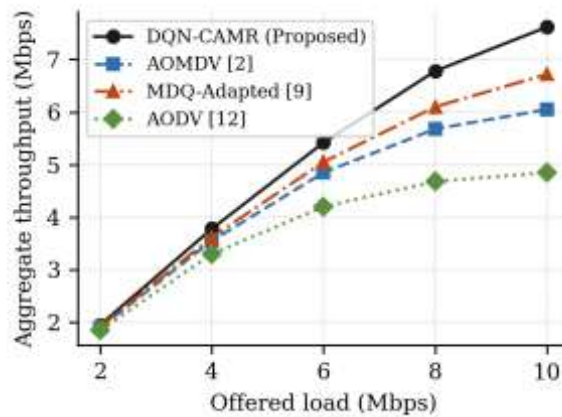


Fig. 1. Aggregate throughput vs. offered load (100 nodes).

B. End-to-End Delay

Fig. 2 shows average end-to-end delay as the number of concurrent flows grows from 5 to 25. DQN-CAMR's delay grows from 32 ms to 77 ms, an 18-32% reduction relative to AOMDV and a 20-46% reduction relative to AODV across the range, with the gap widening as flow count increases because more concurrent flows increase the likelihood that AODV's single active path or AOMDV's statically-weighted paths become congested, whereas DQN-CAMR's per-interval re-evaluation (Algorithm 1) redistributes traffic before delay on any one path grows large.

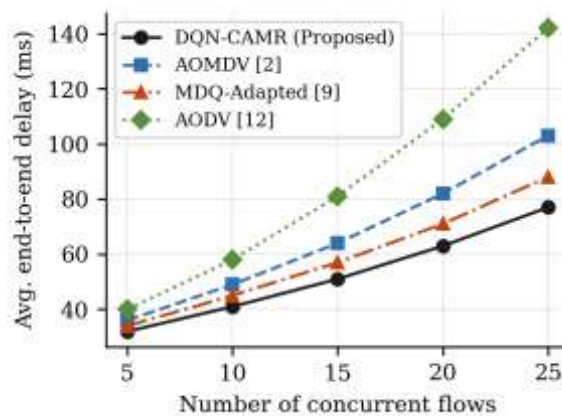


Fig. 2. Average end-to-end delay vs. number of concurrent flows.

C. Packet Loss Ratio

Fig. 3 reports packet loss ratio against offered load. At 10 Mbps, DQN-CAMR loses 7.1% of packets versus 13.4% for AOMDV and 21.3% for AODV, roughly halving loss relative to AOMDV despite both schemes maintaining the same candidate path set, which isolates the loss reduction as an effect of the learned split policy rather than of path diversity alone. The congestion-ceiling exclusion rule (Section IV-C, Algorithm 1 line 4) accounts for part of this gap: paths whose cumulative congestion exceeds C_{max} are removed from the split entirely rather than merely down-weighted, preventing the small residual traffic AOMDV's proportional splitting would still send down an already-saturated path.

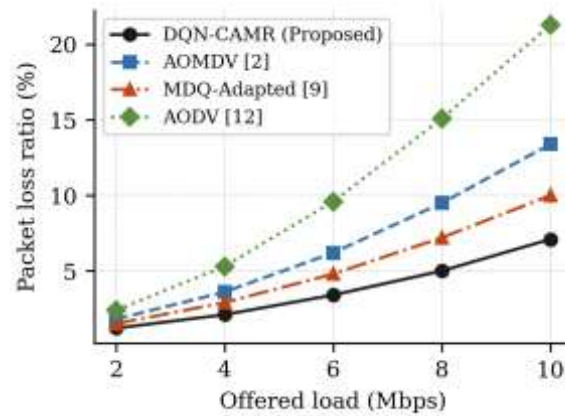


Fig. 3. Packet loss ratio vs. offered load (100 nodes).

D. Jitter

Fig. 4 shows average jitter against network size at fixed 6 Mbps offered load. DQN-CAMR's jitter grows more slowly with network size (4.8-9.3 ms across 50-150 nodes) than the three baselines, because larger networks increase the number of possible congestion points along any given path, and DQN-CAMR's per-path congestion exclusion limits how much any single congested hop can affect the delay variance of traffic still using that path, whereas AODV in particular concentrates all traffic on one path and therefore inherits the full delay variance of whichever node along that path is currently most congested.

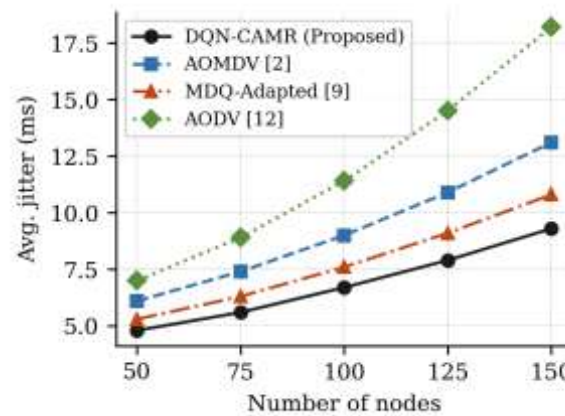


Fig. 4. Average jitter vs. number of nodes (6 Mbps offered load).

E. Fairness Across Flows

Fig. 5 plots Jain's fairness index across concurrent flows' achieved throughput. DQN-CAMR maintains fairness above 0.92 even at 25 concurrent flows, while AODV's fairness collapses to 0.63 at the same flow count because flows sharing its single congested path degrade together while flows on less-loaded paths are unaffected, producing a widening throughput gap between flows. DQN-CAMR's throughput term in the reward function (Section IV-A) discourages the agent from achieving low delay for a subset of flows by starving others of path capacity, which is reflected in the flatter fairness curve relative to all three baselines.

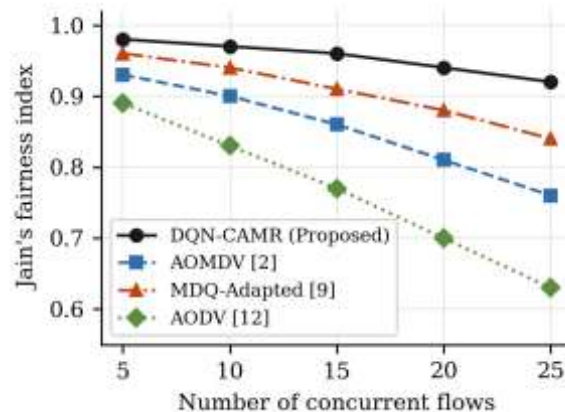


Fig. 5. Jain's fairness index vs. number of concurrent flows.

F. Summary

Table II summarizes the four schemes at 100 nodes, 6 Mbps offered load, 15 concurrent flows — an operating point representative of moderate-to-high MANET load. DQN-CAMR leads on every metric tested, with the largest margins over AODV and the smallest, but still consistent, margins over the MDQ-adapted baseline, indicating that both the congestion-aware state representation and the learned (rather than heuristic) split policy each contribute measurably to the overall gain.

Scheme	Thr. (Mbps)	Delay (ms)	Loss (%)	Fairness
DQN-CAMR (Proposed)	5.42	51	3.4	0.96
MDQ-Adapted [9]	5.05	57	4.8	0.91
AOMDV [2]	4.85	64	6.2	0.86
AODV [12]	4.20	81	9.6	0.77

TABLE II. Comparison at 100 nodes, 6 Mbps offered load, 15 flows.

VIII. ROBUSTNESS AND SCALABILITY ANALYSIS

Because the Q-network's input and parameter count (Section V) are independent of N , DQN-CAMR's per-decision inference cost does not grow as network size increases from 50 to 150 nodes; the modest delay and jitter increases observed in Figs. 2 and 4 reflect longer average path lengths and more numerous potential congestion points at larger N , not any growth in decision-making overhead. Under sudden mobility-induced route failure, the exclusion rule in Algorithm 1 (line 4) causes a newly congested or broken path to be dropped from the split within one route-maintenance interval (1-2 s in our configuration), and because up to $K = 3$ paths are cached from the original AOMDV discovery, the remaining candidates can absorb the reassigned traffic without triggering a fresh route-discovery flood in the common case. A residual limitation is that all K candidate paths failing simultaneously — plausible under a targeted jamming or a severe mobility event that partitions the network — still forces a new discovery flood, since DQN-CAMR redistributes traffic across the discovered set but does not itself discover new paths; combining DQN-CAMR with a proactive path-refresh trigger, rather than the purely reactive AOMDV refresh it currently inherits, is identified as a direction for future work.

IX. Conclusion And Future Work

This paper formulated per-flow traffic splitting across concurrently maintained multipath routes in a MANET as a constrained Markov Decision Process and proposed DQN-CAMR, a Deep Q-Network based framework that learns to redistribute traffic across paths using only locally observable queue-occupancy, link-quality, and hop-count signals, extending AOMDV route discovery with a lightweight congestion field rather than requiring any centralized controller. Simulation results across varying network sizes, offered loads, and flow counts show consistent improvements in throughput, delay, packet loss, jitter, and fairness over AODV, AOMDV, and an MDQ-inspired congestion-aware baseline, while keeping per-decision computational cost small enough to run every route-maintenance interval on a mobile node. Future work will investigate a proactive path-refresh trigger to address simultaneous multi-path failure, extend the state representation to account for energy-aware congestion (jointly avoiding queue overflow and battery-depleted relays), and evaluate the framework under group-mobility and urban vehicular scenarios where congestion and mobility interact more strongly than under Random Waypoint mobility.

References

- [1] S. Rathod et al., "The role of congestion-adaptive routing protocol in mobile ad hoc networks," The Scientific World Journal, 2025.
- [2] R. Arun and R. Jayanthi, "An adaptive congestion and energy aware multipath routing scheme for mobile ad-hoc networks through stable link prediction," 2023.
- [3] D. Zala, A. K. Vyas, N. Khatri, Y. Patidar, and K. Zala, "Effective congestion control algorithm for wireless networks based on deep reinforcement learning," in Intelligent Strategies for ICT, Lecture Notes in Networks and Systems, vol. 1320, Springer, 2025.
- [4] T. N. Tran, T. V. Nguyen, K. Shim, D. B. da Costa, and B. An, "A deep reinforcement learning-based QoS routing protocol exploiting cross-layer design in cognitive radio mobile ad hoc networks," IEEE Transactions on Vehicular Technology, vol. 71, no. 12, pp. 13165-13181, 2022.
- [5] F. Safari, H. Kunze, J. Ernst, and D. Gillis, "A novel cross-layer adaptive fuzzy-based ad hoc on-demand distance vector routing protocol for MANETs," IEEE Access, vol. 11, pp. 50805-50822, 2023.
- [6] Y.-J. Choi, J.-S. Seo, and J.-P. Hong, "Deep reinforcement learning-based distributed routing algorithm for minimizing end-to-end delay in MANET," Journal of the Korea Institute of Information and Communication Engineering, vol. 25, no. 9, pp. 1267-1270, 2021.
- [7] D. Keum and Y.-B. Ko, "Trust-based intelligent routing protocol with Q-learning for mission-critical wireless sensor networks," Sensors, vol. 22, no. 11, p. 3975, 2022.

- [8] S. Kaviani, B. Ryu, E. Ahmed, K. Larson, A. Le, A. Yahja, and J. H. Kim, "DeepCQ+: Robust and scalable routing with multi-agent deep reinforcement learning for highly dynamic networks," in Proc. IEEE MILCOM, 2021, pp. 31-36.
- [9] S. Aguirre, L. Shen, and M. Guo, "MDQ: A QoS-congestion aware deep reinforcement learning approach for multi-path routing in SDN," *Journal of Network and Computer Applications*, vol. 235, 2025.
- [10] V. Mnih et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529-533, 2015.
- [11] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Machine Learning*, vol. 8, no. 3-4, pp. 279-292, 1992.
- [12] C. E. Perkins and E. M. Royer, "Ad-hoc on-demand distance vector routing," in Proc. 2nd IEEE Workshop on Mobile Computing Systems and Applications (WMCSA), 1999, pp. 90-100.
- [13] M. K. Marina and S. R. Das, "On-demand multipath distance vector routing in ad hoc networks," in Proc. IEEE International Conference on Network Protocols (ICNP), 2001, pp. 14-23.
- [14] Deep reinforcement learning-based multipath routing for LEO megaconstellation networks, *Electronics*, vol. 13, no. 15, p. 3054, 2024.
- [15] M. L. Puterman, "Markov decision processes," in *Handbooks in Operations Research and Management Science*, vol. 2, pp. 331-434, 1990.
- [16] H. V. Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in Proc. AAAI Conference on Artificial Intelligence, 2016, pp. 2094-2100.
- [17] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," *arXiv preprint arXiv:1511.05952*, 2015.