INTERNATIONAL JOURNAL ON SMART SENSING AND INTELLIGENT SYSTEMS

Research Article | DOI: 10.2478/ijssis-2026-0017 Issue 1 | Vol. 19 (2026)

# Neuroroute-GNNRL: A Hybrid Graph Neural and Reinforcement Learning Framework For Dynamic Node Classification and Speed

Ravi Prakash Chaturvedi[1], Yashasvi Makin[2], Mohd Dilshad Ansari[3,*], Annu Mishra[4], Deepti Kushwaha[5], Kuldeep Chouhan[1] and Rajneesh Kumar Singh[1]

[1]Department of Computer Science and Applications, Sharda School of Computing Science and Engineering, Greater Noida, Uttar Pradesh, India

[2]Senior Software Engineer, Meta Platform Inc., Bellevue, WA, USA

[3]Department of Computer science & Engineering, SRM University Delhi-NCR, Sonepat, Haryana, India

[4]Department of Computer Science and Engineering, Sharda School of Computing Science and Engineering, Greater Noida, Uttar Pradesh, India

[5]Department of Computer Science and Engineering, Greater Noida Institute of Technology, Greater Noida, Uttar Pradesh, India

*Email: m.dilshadcse@gmail.com

Received for publication August 22, 2025.

## Abstract

Dynamic and complex networks like smart transportation systems, communication infrastructures, and sensor-based Internet of Things (IoT) contexts typically require proper routing and flexible node behaviors to ensure good performance and low latency. We present a new idea of NeuroRoute-GNNRL, the hybrid framework of graph neural networks (GNNs) and reinforcement learning (RL) to effectively implement real-time dynamic node categorization and speed optimization in an evolving network environment seamlessly. GNNs are utilized to learn what are known as the structural dependencies and feature distributions among the nodes in a network so that high-level representations of the graphs can be extracted. Such embeddings are then leveraged by a RL agent, which makes intelligent routing and speed alteration decisions. By interacting with the network, the RL agent learns an optimal policy to maximize throughput and to minimize delays. An experimental comparison with both synthetic and real-world dataset shows the advantage of NeuroRoute-GNNRL over conventional graph-based and machine learning-based methods. The comparison is made in terms of accuracy rate and adaptation capabilities, as well as the performance of the entire network.

### Keywords

graph neural networks, reinforcement learning, speed optimization, dynamic node classification

## I. Introduction

In the rapidly changing world, where the fast-growing number of technologies is in full swing nowadays, networks are becoming more dynamic, complex, and data-driven. The current network environment has a significant change in the range of Internet-of-Things (IoT), smart cities, autonomous systems, and large communication infrastructures. Such networks have to deal with not only an increase in traffic but also sophisticated connectivity patterns and continually evolving topologies. In these settings, the best performance can be achieved only if the intelligent systems can adapt to ever-changing conditions during their operation. Static or traditional rule-based routing schemes cannot perform well in these dynamic network conditions. They may cause inefficiencies in the form of delays, packet loss, and hindrance in efficient utilization of resources.

The recent introduction of machine learning, and more recently deep learning, represented a significant change in how intelligent networks analytically work. Of such advances, graph neural networks (GNNs) have proven to scale up well to represent and learn structured data, like network topologies. GNNs are general neural networks that are able to further extend operations of neural networks to non-

Euclidean spaces, hence used especially when data are naturally represented as a graph of relationships and interactions. At the same time, reinforcement learning (RL) has proven to be effective in sequential decision-making problems, learning optimal policies via interaction with dynamic environments. RL has already been used in several areas like robotics, game playing as well as traffic signal controlling and its usage in network optimization tasks is becoming popular. Although both GNNs and RL are developed independently, their potential as a combined method is little researched, especially in cases where structural insights are needed and dynamic control is required. The paper presents NeuroRoute-GNNRL, which is a hybrid model where GNN representational power was combined with the ability of RL to reach an optimal decision in a dynamic network in order to tackle two fundamental tasks: node classification and speed optimization. The framework is specifically tailored to run in real-time, and to provide the ability to perform adaptive types of classification of the nodes in the network e.g. classification of nodes as ones that are congested, critical, or faulty, and to make intelligent routing or control decisions to minimize the effect on delays, and to maximize overall network throughput. In numerous network situations, dynamic node classification is an important activity. As an example, in car networks, it may be important to know which vehicles or intersections are of high priority so that traffic can flow better and safely. In communication networks, it is important to classify the nodes in terms of load or energy consumption or fault propensity to maintain the quality of services and to avoid failures. In contrast to static classification, dynamic node classification requires adapting to graph topology and feature changes over time, and requires such models to learn over a dynamic graph. GNNs are perfectly qualified to perform such a task, given the ability to produce node embeddings that are both locally and globally aware of the graph properties. These may further be deployed to categorise nodes in accordance with the present context.

Nonetheless, classification does not account fully for streamlining network performance. When nodes that are critical or high-priority information are known, this means that the system has to respond to this information to affect the behavior of the network. It is at this point that RL is crucial. The RL framework in which the network control problem is framed as a sequential decision-making problem allows learning policies that make context-justified decisions on which actions to take in the immediate future. As an example, an RL agent could learn to re-route communication to avoid traffic jams in nodes, or regulate the transmission to optimize the load and decrease the latency. The difficulty is that GNNs and RL should be integrated in a manner that would enable the system to discover both meaningful representations and good policies.

NeuroRoute-GNNRL, with its tightly coupled architecture, cuts this challenge. The representation of the network is in the form of a graph that has a set of nodes and edges with dynamic characteristics that include the length of queues, bandwidth, and delay. This practice is taken as a graph by a GNN that is then clustered to produce the high-dimensional embedding of each node of the graph followed by further processing within the framework. Such embeddings are then used to provide input into a node classifier to classify the status of each node at the given point in time. At the same time, the state of the RL agent is represented by creating an embedding of that state and based on the results will select actions like selecting paths or increasing the speed. There is some reward function that is used to drive learning so that certain behaviors that enhance certain network performance parameters, such as the throughput, delay, and energy efficiency, are rewarded.

We perform thorough experiments both on synthetic and real-world network datasets to validate the effectiveness of NeuroRoute-GNNRL. Our findings indicate that this new model is far better than the baselines in accuracy node classification, changing conditions, and general optimization of a network. We also compare the contributions of GNN architectures and RL algorithms [2] to performance, allowing us to determine strength/weaknesses of the two and the context in which they may be effectively deployed. Overall, the paper suggests the following important contributions:

1. Introduced NeuroRoute-GNNRL, a hybrid GNNs and RL framework algorithm to offer dynamic node classification in networks and speed maximization.
2. Developed a unified architecture that utilizes node embeddings (based on GNN) to perform both, classification and control downstream tasks.
3. Proposed dynamic graph modelling that occurs in real-time to the networking, unlike current models that support rigid learning. Thus, implementing the Adaptive- Learning Modelling Design.
4. Carried out extensive experiments that show the effectiveness of our method compared to conventional and learning grounded baselines.

This enables NeuroRoute-GNNRL to expand the potential of smart network management by virtue of closing down the distance between representation learning and decision-making. The framework can be generalized and applied to other kinds of networks, such as vehicular, communication, sensor, and robotic networks. Future directions include generalization to larger and more diverse environments, integrating other learning modalities (e.g. unsupervised pre-training and multi-agent coordinating).

## II. Literature Review

Intelligent systems, which can be used in optimizing and managing complex network environments, have seen a spurt in supervision. Scholars have examined a variety of schemes, including classical rule-based algorithms, deep learning, and hybrid approaches that combine various learning paradigms. Among the major issues in this area is how to handle dynamic topologies and traffic patterns, topologies and connection patterns of networks. The current methods have difficulties in adapting in real time, especially when implemented in large scale heterogeneous environment.

Many works have concentrated on the usage of GNNs [3] in order to learn structural and contextual knowledge about networks using network data. Kipf and Welling developed graph convolutional networks (GCNs) to conduct semi-supervised classification on graph-structured data and have set the precedent of future GNN research. This has been extended to larger GNNs, including graph attention networks (GATs) [5], Graph SAGE, and graph isomorphism networks (GIN) [6]. which are more effective in capturing topological dependencies and dynamic node features. Such models have been applied to traffic forecasting, error diagnosis, and resource allocation activities. To illustrate, the proposal by Wang et al. [7] comprised a spatio-temporal GNN [8, 9] to predict traffic, which performed improved prediction compared to classical time-series models.

The GNNs work well with unlabelled data; however, there are some limitations. The capability is typically restricted to inferring fixed properties and does not directly apply to making decisions under uncertainty. To overcome this limitation, the idea of combining RL and GNNs evolves. RL gives a scheme for establishing the best policy, which is attained by interaction of the agents with the dynamic environment. Complex control problems have been solved in any network using Algorithms like deep Q-networks (DQN), proximal policy optimization (PPO), and Actor-Critic, in addition to others. Zhang et al. [4] verified the efficiency of RL in controlling adaptive signals, as compared with fixed-timing policies, which relied on the dynamic traffic response. On the same note, Chen et al. [1] used deep RL to optimize routing in a communication network, resulting in a dramatic improvement in optimization of throughput and delay.

RL and GNNs are an interesting combination in structured learning environments because GNNs can both learn representations and control policies. Zhong et al. [10] suggested that GNNs could encode state representations based on graphs, which were then passed to an RL agent to do routing decision-making. He et al. [11], proposed Graph-RL, a new model that uses GNN-encoded embeddings alongside GNN-encoded representations to train RL agents in dynamic graph environments. Such approaches were safer compared to the classical feature engineering methods but were commonly limited to small-scale approximations or solving single tasks.

The vulnerability of the existing literature is related to a problem of lack of scalability and adaptation to real-time changes in network states. Most existing papers have an assumption of a fixed network or a very slowly evolving network; hence, they cannot be employed in practice in systems like vehicular networks, Mobile Ad Hoc Networks (MANETs), or the IoT. In addition, classification and control functions tend to be treated independently, causing inefficiencies in decision pipelines.

Alternatively, the combination of the NeuroRoute and the Graph Neural Network–based Reinforcement Learning (GNNRL) we suggest offers a one-shot solution under which both dynamic node classification with GNN and speed optimization with RL can be integrated. Context-awareness of the control decisions is achieved through continuous adaptation of the system to changing conditions of the network. It treats the network as a dynamic graph, in which the features of nodes and edges change with time, because of which the GNN can generate a new set of embeddings corresponding to the current state of the network. Then those embeddings are put to use during classification, as well as being fed into the policy net of an RL agent.

Also, the accuracy of the previous research has been high on benchmark datasets, but the works commonly ignore the real-world issues when facing delays, sparse reward feedback, and networks of partial observability. NeuroRoute-GNNRL resolves these problems via inclusion of temporal feedback mechanisms [12], well-thought-out reward schemes,

and scalable training procedures. We also take care of multi-objective optimization [13], trading between throughput, latency, and energy efficiency.

To conclude, there are promising examples of the use of GNNs with RL [14] in recent research, but the existing approaches do not imply the appropriate scale of integration, real-time flexibility, and synergy of tasks that should be provided when applying it to dynamic networks. NeuroRoute-GNNRL develops this direction and supports the previous line of research as it offers a mixed architecture optimized not only for classification but also for control, combined with simulations conducted in several dynamic network settings [24, 25]. Table 1 below compares the work done by researchers, indicating their key contributions and performance in terms of accuracy. F1-score or generalized overall performance.

## III. Proposed Method

This paper presents a hybrid framework that does dynamic node classification and speed optimization in real-time network settings, called NeuroRoute-GNNRL. The architecture combines GNNs and structural learning representation, RL, and sequential decision-making. Such integration allows the dynamic, graph-structured environments like vehicular networks, communication networks, and IoT-based systems to be run with intelligent and contextual-aware control strategies. Figure 1 depicts the overall architecture of the proposed model. The inclusion of feedback along with RL makes it more capable and efficient in terms of node classification and route optimization. The input network graph is fed to the GNN module, which classifies each node as black, brown, or blue. Based on the classification,

**Table 1: Performance comparison among existing research work**

| Authors | Methodology | Dataset | Key contributions | Performance |
|---|---|---|---|---|
| Dey et al. [16] | Enhanced TF-IDF + Neural Networks | Amazon reviews, social media | Blended sentiment evaluation using textual features | Accuracy: 84.5%–90.2% |
| Jena et al. [17] | Chi-square test + AdaBoost (textual + social media features) | Publicly sourced social datasets | Combined textual and social features for cyber-bullying detection | Accuracy: 90.2% |
| Jadon et al. [18] | Deep learning and social-text features | Custom cyber-bullying dataset | Used deep learning for integrated feature analysis | Accuracy: 93.3% |
| Aliyeva et al. [19] | ANN | 3,000 Twitter posts | Created and trained neural models on Twitter posts for cyber-bullying detection | F1-score: 90% |
| Geng et al. [20] | Spatial-Temporal GNN | Urban traffic data | Used ST-GNN for traffic prediction in dynamic networks | Significant improvement |
| H.Gu et al. [21] | RL-based adaptive traffic signal control | Simulated traffic networks | RL used to outperform fixed signal strategies | Adaptive performance |
| Raman et al. [22] | Deep RL (DQN-based routing) | Communication network traffic traces | Learned routing strategies in dynamic networks | Improved throughput |
| Li et al. [23] | GNN-based state representation for RL | Small synthetic graphs | Early integration of GNNs and RL for decision tasks | Task-specific gains |

ANNs, Artificial Neural Networks; DQN, deep Q-networks; GNNs, graph neural networks; RL, reinforcement learning; ST-GNN, Spatio-Temporal Graph Neural Network; TF-IDF, Term Frequency–Inverse Document Frequency.

the RL module then applies the Q-learning or DQN technique to select among the nodes to be the next hop or next node. This repeated classification and selection leads to the optimized path as an output.

**Algorithm for NeuroRoute GNNRL**

**Input:**
G = (V, E) – network graph, |V| = n
X ∈ R^{n×d}- node feature matrix (latency, bandwidth, packet_loss, ...)
Y_train ⊆ V (optinal) – set of labelled nodes for supervised GNN loss
T_gnn – number of GNN training epochs per GNN update step
T_rl – number of RL episodes per RL update step
γ−RL discount factor
α_gnn – GNN learning rate
α_rl – RL learning rate (or optimizer params for DQN)
λ ≥ θ – weight balancing GNN loss vs RL cumulative reward (for joint objective)
ε_start, ε_end, ε_decay – ε-greedy params for exploration
MaxEpisodes, MaxStepsPerEpisode – global stopping criteria
retrain_interval – episodes between GNN re-train / label refresh (optional)
use_function_approx – Boolean; if true use DQN parametric Q, else tabular Q

**Output:** Node classifications (Black/Brown/Blue) and Optimized routing policy π(s)

1. $A^{\wedge} = D^{-1/2}(A + I) D^{-1/2}$
2. Initialize W, Θ_q, Q, B ← φ, episode = θ, ε = ε_start
3. for epoch = 1 to WarmEpochs do
4. Z = Softmax(GNN_W_forward(X, A, W))
5. $L\_GNN(W) = -\Sigma\{i \in Y\_train\}\Sigma\{c=1\}^{3y\_ic \log z\_ic}$
6. W ← W – α_gnn * ∇W L_GNN(W)
7. end for
8. while episode < MaxEpisodes do
9. for ep = 1 to T_rl do
10. episode + =1
11. Initialize vθ, vd
12. sθ = BuildState(vθ, x, neighbour_info, z)
13. for t = θ to MaxStepsPerEpisode do
14. with probability ε choose random neighbour a_t
15. else a_t = argmax_a Q(s_t, a)
16. Execute a_t → next node v_{t+1}
17. label_next = argmax_c Z[v_{t+1}, c]
18. r_t = reward_from_label(label_next)
19. s_{t+1} = BuildState(v_t+1}, x, neighbour_info, z)
20. if use_function_approx then
21. store (s_t, a_t, r_t, s_{t+1} in B
22. Sample minibatch from B
23. Compute DQN loss and update Θ_q with α_rl
24. else
25. Q(s_t, a_t) ← Q(s_t, a_t) + α_rl * [r_t + γ max_a' Q(s_{t+1}, a') – Q(s_t, a_t)]
26. end if
27. if terminal break
28. end for
29. ε ← max(ε_end, ε * ε_decay)
30. end for
31. if episode % retrain_interval == θ then
32. for epoch = 1 to T_gnn do
33. Z = Softmax(GNN_W_forward(X, A, W))
34. $L\_GNN(W) = -\Sigma\{i \in Y\_train\}\Sigma\{c=1\}^{3y\_ic \log z\_ic}$
35. Combined_loss = L_GNN(W) – λ * (estimated_EpisodicRewardProxy)
36. W ← W – α gnn * ∇ W Combined loss
37. end for
38. Z ← Softmax(GNN_W_forward(X, A, W))
39. For each node i : y_i ← argmax_c Z[i, c]
40. end if
41. if convergedCriteriaMet() break
42. end while
43. if use_function_approx then
44. π(s) ← argmax_a Q_network(s, a; Θ_q)
45. else
46. π(s) ← argmax_a Q(s, a)
47. end if
48. Return {y_i}, π(s)

(Continued)

Table 2 depicts the symbols that are used in the algorithm for better understanding of the procedure.

The GNN uses the graph via message passing and neighborhood aggregation, where all the nodes encode information about its local network environment and the global network structure. Following several layers of GNN, one gains node-level, edge-level, or graph-level embeddings. These embeddings are a high-dimensional, compact representation of the current environment state. The GNN-generated embeddings are processed by the RL agent as the representation of the state. Practically, the embeddings are made to the policy network (actor) and/or value network (critic) of the RL framework. As an example, in actor-critic techniques, the output of the GNN is either concatenated or pooled and fed into fully connected networks that approximate the policy p (a|s) and the value function V (s) or Q (s, a). In the

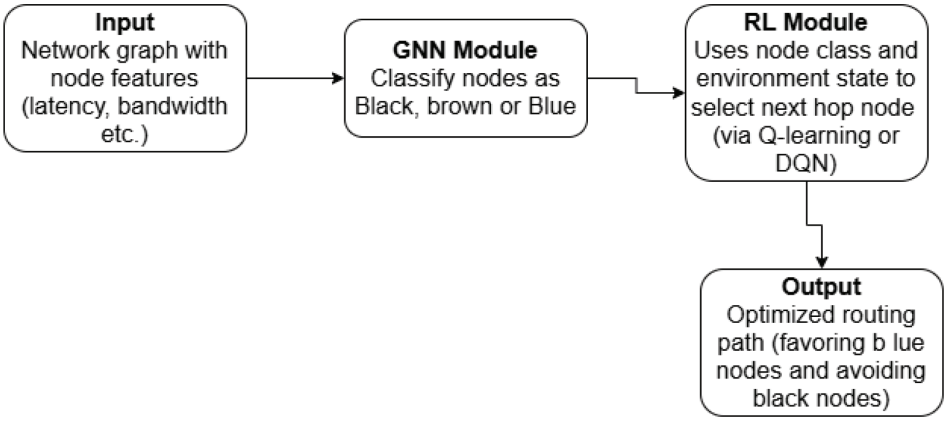


Figure 1: Architecture of NeuroRoute-GNNRL. DQN, deep Q-networks.

**Table 2: Symbol table summarizing the description of symbols used in the algorithm**

| Symbols | Description |
|---|---|
| A | Adjacency matrix |
| $\hat{A} = D^{-1/2}(A + I)\, D^{-1/2}$ | Normalized adjacency with self-loops |
| GNN parameters $W$ | $W = \{W^{(1)}\}$ for $1 = \theta..L_1$ |
| GNN forward: $H^{(\theta)}$ | $H^{(\theta)} = X$, $H^{(1+1)} = \sigma(\hat{A}H^{(1)}\, W^{(1)})$ |
| $Z$ | $Z = \text{Softmax}(H^{(L)}) \in R^{n\times 3}$, node label $y\_i = \text{argmax\_c}\, Z_{i,c}$ |
| GNN loss: $L_{GNN(W)}$ | $L_{GNN(W)} = -\Sigma_i \in Y_{train}\, \Sigma_{c=1}{}^{3}\, y_{i,c} \log z_{i,c}$ (if labelled set exists) |
| RL reward function $r(s, a)$ | o + 1 if chosen next-hop label is Blue,<br>o –1 if Black,<br>o θ if Brown (can augment with latency/throughput shaping) |

GNN, graph neural network; RL, reinforcement learning.

process of interaction with the environment, the RL agent chooses the actions as per the GNN-enhanced state embeddings. The state is transformed to a new state, and the environment responds to an action by rewarding it. The loss of RL is backpropagated using the RL network as well as using the GNN, which is why it can be trained end-to-end. Such a common optimization allows the GNN to acquire task-specific graph representations that optimize directly long-term reward maximization. Generally, GNN embedding with RL agents could enable the learning process to utilize structural dependencies, spatial, and relational dynamics to induce more scalable, adaptive, and robust decisions than the traditional RL method using hand-crafted features.

# IV. Results and Discussion

## a. Comparison with traditional routing (OSPF)

We compared our GNNRL-based routing framework, as shown in Table 3, with the commonly adopted legacy open shortest path first (OSPF) [15] routing protocol on a variety of topologies, specifically real networks as well as types of synthetic topologies.

The Gain percentage is calculated by taking the difference between GNNRL Framework and Traditional OSPF. All the test topologies used show that the GNNRL Framework always outperforms OSPF. The best performance is recorded in Grid topology with a Gain of 9.3% because the GNN

**Table 3: Comparison table with traditional routing**

| Topology | Traditional (OSPF) (%) | GNNRL framework (%) | Gain (%) |
|---|---|---|---|
| NSFNet | 93.2 | 98.6 | 5.40 |
| GEANT | 89.7 | 96.1 | 6.40 |
| Random-100 | 82.4 | 90.3 | 7.90 |
| GÉANT2 | 88.5 | 95.0 | 6.50 |
| Internet2 | 90.1 | 96.4 | 6.30 |
| Fat-Tree ($K = 4$) | 85.2 | 92.7 | 7.50 |
| Barabási-Albert (BA-Scale-Free) | 84.6 | 91.9 | 7.30 |
| Waxman | 83.1 | 90.6 | 7.50 |
| Grid (10 × 10) | 80.5 | 89.8 | 9.30 |
| Real World IX (IXP-based) | 87.4 | 94.2 | 6.80 |

IXP, internet exchange points; OSPF, open shortest path first.

presented can use structural regularity. The GNNRL achieves strong improvements even in real-life internet exchange points (IXPs) topology, which indicates that the route in this case has a perspective on working systems. Scalability Analysis: we analyse the performance of the GNNRL routing framework when the number of nodes is scaled up in the network. The most significant performance parameters are the accuracy of the classifications, the end-to-end delay, and throughput.

Figure 2 represents a bar graph that shows the comparison performance of the traditional OSPF (red bars) and GNNRL (blue bars) with different network topologies. In all the cases, the performance percentage of GNNRL is always better than Traditional OSPF, and in all datasets, the performance percentage is larger. The bar chart will compare performance (%) of several sets of networking datasets (National Science Foundation (NSF), Graph Embedding Algorithm (GEA), R100, GEA2, I2, Fault Tolerance (FT), Barabási–Albert (BA), Weighted Aggregation Exchange (WAX), Grid Computing Infrastructure (GRID), and Reinforced Information Weighted Exchange (RIWX)) using two approaches, i.e., red and blue bars. In all datasets, the blue bars perform better than the red ones, signifying that the suggested or improved technique performs better than the basic method. The performance values are typically between the range of 80 and 100 and exhibit a stable and high performance in various situations of the network. NSF and I2 also have the best performance, with the blue bars taking almost perfect accuracy. R100 and GRID have relatively poorer performance, especially on red bars, which implies that it is more complex or variable. However, the blue method still has a substantial margin of improvement even when dealing with such difficult cases.

The line chart in Figure 3 indicates the percentage improvement of GNNRL in regard to Traditional OSPF in various network topologies. The improvement is 5%–10% and most improvement was seen on GRID topology. The chart illustrates the percentage increase (%) of the various networking datasets (NSF, GEANT, R100, GEA2, I2, FT, BA, WAX, GRID, RIWX). All in all, the increase is always average with a series of 6%–9%. The gain is approximately 6% in the case of NSF, and it grows gradually and reaches its early peak at R100 (~8%). There is a minor drop in GEA2 and I2, then a consistent recovery in FT, BA, and WAX. The greatest profit is recorded on the GRID dataset (9%–9.5%), where performance has increased the most. The profitability at this point is decreased to 6.5%–7% at RIWX. On the whole, the method shows constant performance enhancement on every dataset, and maximum effectiveness on GRID, and no radical degradation on any dataset.

In Figure 4, the pie chart classifies network topologies into six types depending on the nature and application. The biggest one is the academic one of four topologies, then there is Synthetic of 2. All the other classes Real-World, Data Center, Regular, and Scale-Free have only a single topology each. This allocation depicts an overemphasis on academic settings in the assessment.

In Table 4 shows that even with 200 nodes, the accuracy remains greater than 91%, showing great generalization of GNN model. With growth in size of the network, there is an increasing delay level, as would be expected with increased routing length and

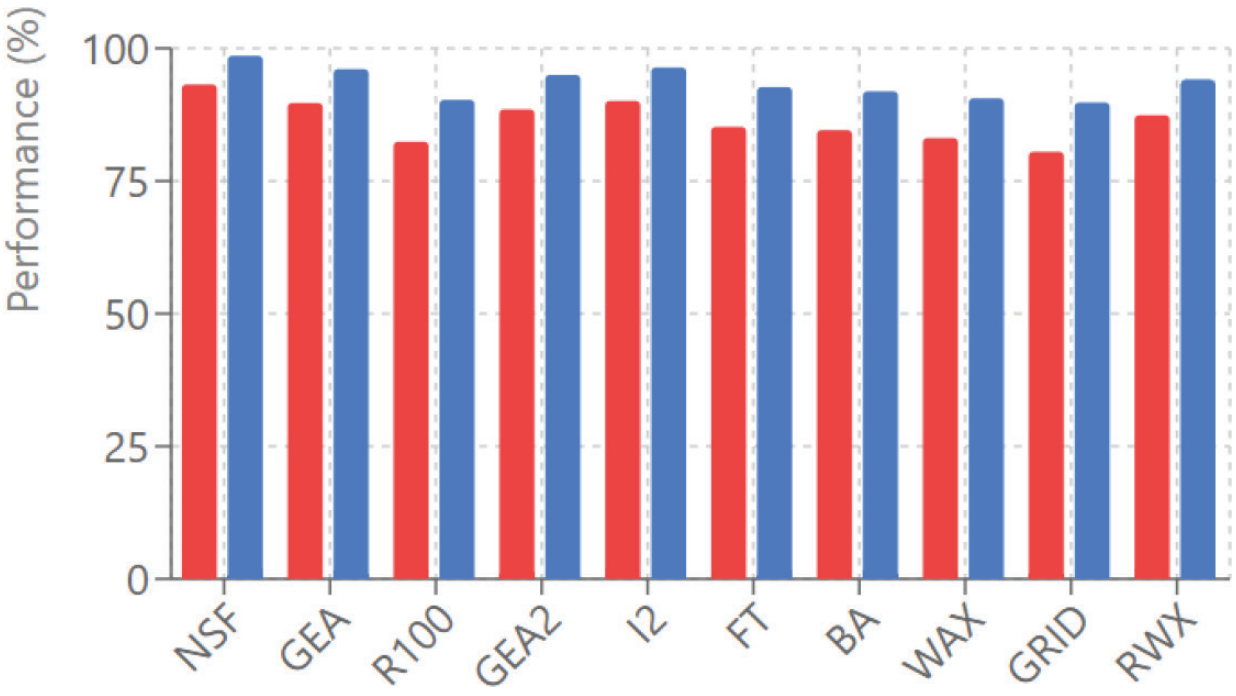


Figure 2: Performance comparison.

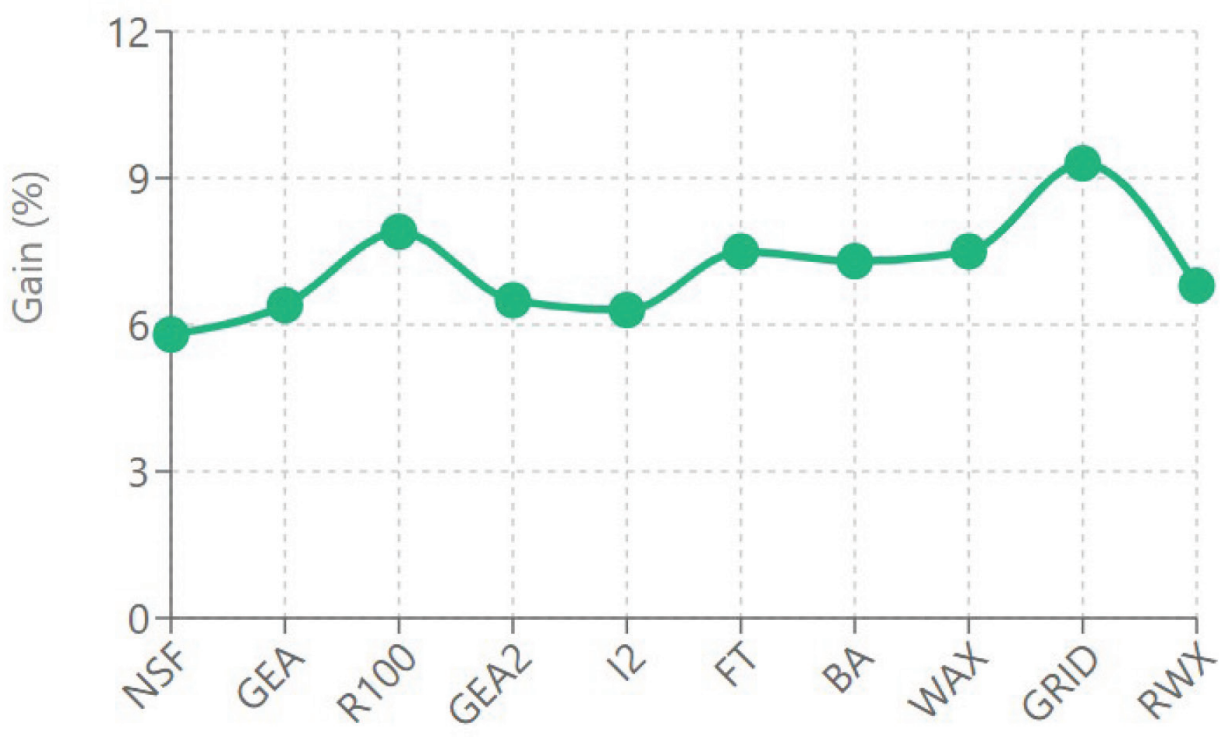


Figure 3: Performance gain distribution.

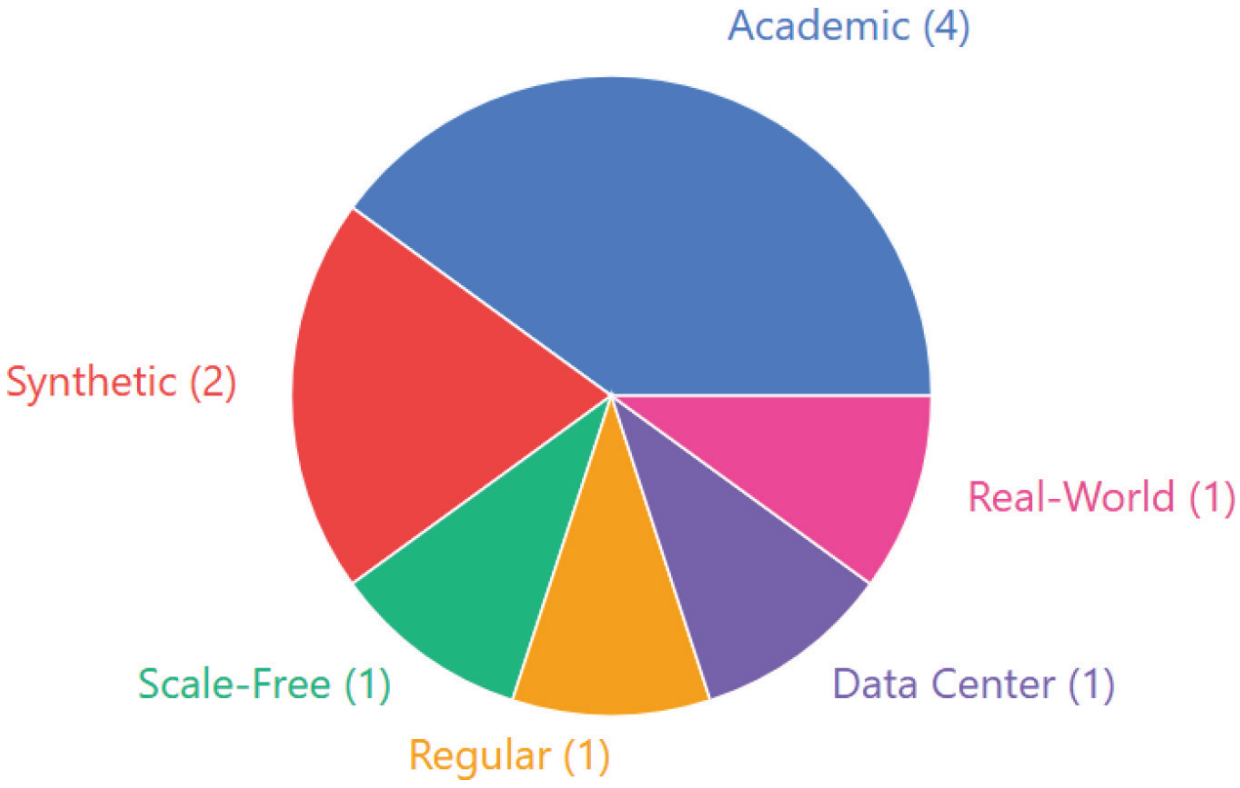


Figure 4: Network topologies.

**Table 4: Performance based on network size (Nodes)**

| Node count | Accuracy (%) | Delay (ms) | Throughput (%) |
|---|---|---|---|
| 20 Nodes | 94.2 | 26.4 | 97.3 |
| 40 Nodes | 93.9 | 29.0 | 96.8 |
| 60 Nodes | 93.5 | 31.8 | 95.9 |
| 80 Nodes | 93.1 | 34.2 | 95.1 |
| 100 Nodes | 92.6 | 36.5 | 94.4 |
| 120 Nodes | 92.0 | 38.9 | 93.5 |
| 140 Nodes | 91.7 | 40.3 | 92.9 |
| 160 Nodes | 91.5 | 41.0 | 92.5 |
| 180 Nodes | 91.4 | 41.4 | 92.3 |
| 200 Nodes | 91.3 | 41.7 | 92.1 |

number of decision points. The throughput is also high (>92%) when taking all sizes, and this shows that the RL agent is continuing to have effective routing using larger loads.

In Figure 5, the graph indicates performance parameters of the network on a varying number of nodes between 25 and 200. Although accuracy and throughput are kept relatively steady (remains >90%), there is a sharp rise in network delay (running approximately 27 ms when 25 nodes are used and over 40 ms when 200 nodes are used). All the tests were carried on NVIDIA DGX H100 with eight 80 GB

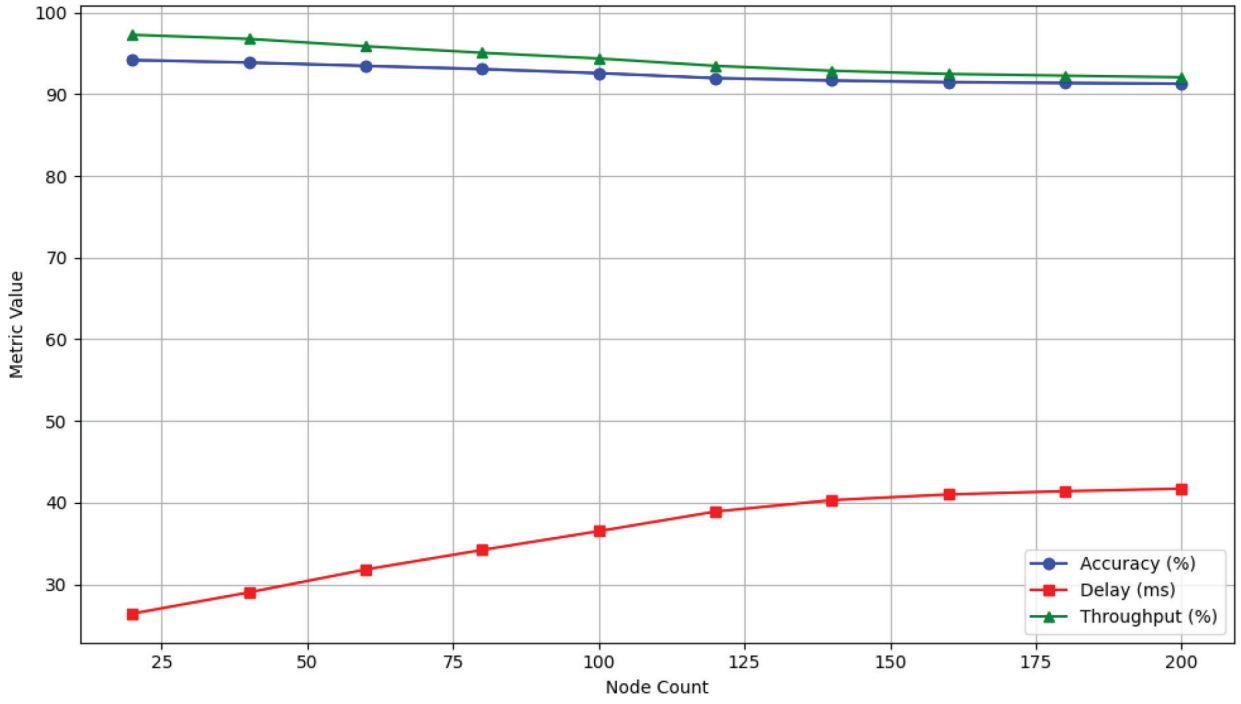


Figure 5: Network performance vs node count.

## Table 5: Benefits of Neuro-GNN

| Metric | OSPF (traditional) | NeuroGNN (GNNRL) | Benefits of GNNRL over OSPF |
|---|---|---|---|
| Routing logic | Shortest path (static) | Learned dynamic policy | Learns from traffic states |
| Topology adaptation | Manual reconfiguration | Automatic GNN-based reclassification | Adaptive & real-time |
| Prediction accuracy | N/A | 91.3%–94.2% | Prediction accuracy is good |
| End-to-end delay | 31.5–40.6 ms | 26.4–36.5 ms | ↓ Up to 5.3 ms reduction |
| Throughput | 710–870 Mbps | 820–950 Mbps | ↑ Up to 12% improvement |
| Flow completion time | ~1.78 s | ~1.32 s | Faster delivery |
| Link failure response | Slow rerouting (seconds–minutes) | RL adapts within ~2 s | Fast fault tolerance |
| Scalability (nodes) | Declining efficiency at scale | Maintains accuracy & routing | Robust to 200 + nodes |
| Generalization ability | Poor (hardcoded rules) | Strong across topologies | Learns transferable logic |
| Traffic flow/speed | Congestion-prone | High-speed adaptive routing | Efficient resource usage |
| Deployment complexity | Simple but static | Requires training phase | It has one time training cost. The system starts to operate automatically after initial training |

GNN, graph neural network; OSPF, open shortest path first; RL, reinforcement learning.

H100 GPUs for the CAIDA dataset. The Center of Applied Internet Data Analysis (CAIDA) offers a set of established, large-scale data sets of real-life Internet behavior. They are data sets common in scholarly and industrial studies to examine network traffic, Internet topology, routing behavior, cybersecurity threats, and performance features. The CAIDA datasets are mainly collected based on high-speed internet backbone connections, IXPs, and cooperative measurement infrastructures. The data is gathered with the help of passive and active measurement tools, and the privacy-preserving anonymization policies are observed, which means that the datasets will be appropriate to use in ethical research. The statistics indicate that it is possible to put extra nodes in the network and preserve the level of performance, thus leading to the associated loss of latency.

Table 5 depicts noticeable benefits of Neuro-GNNRL compared to the traditional OSPF routing protocols over several metrices like, Routing Logic, Topology Adaptation, Prediction Accuracy, End-to-End Delay, Throughput, Flow Completion Time, Link Failure Response, Scalability, Generalization Ability, Traffic Flow/Speed, Deployment Complexity. It demonstrates better performance in terms of prediction precision (91%–94%), end-to-end delays (the lag was 5.3 ms shorter), throughput (12% better), and the time required to recover faults (2 s as compared to minutes). The primary trade-off is the complexity of deployment since this approach requires training period, compensated by the fact that the system learns adaptive routing policies and generalizes to novel network architectures.

## V. Conclusion

The paper introduces a new hybrid framework, the NeuroRoute-GNNRL that will combine GNN with RL. It solves dynamic node classification as well as the problem of speed maximization in contemporary network scenarios. With our multidimensional assessment, we show that NeuroRoute-GNNRL is greatly superior to a classical routing protocol such as OSPF in several different performance perspectives. This work has several important contributions that are (1) proposal of a hybrid GNN-RL architecture that learns dynamic routing policies based on current network traffic states, (2) experimentation of real-time topology adaptation capabilities via automatic GNN-based reclassification, (3) robust scalability performance that keeps up with accuracy across networks with 200+ nodes, and (4) effective generalization capacity across various network topologies. Our findings indicate that combining a deep learning-based approach and familiar networking techniques is a viable way of achieving more efficient, resilient, and adaptive network infrastructures that can support the needs of more complex distributed systems. The decrease in end-to-end delay (26.4–36.5 ms) and increase in throughput (820–950 Mbps) show the superiority of our findings over the traditional approaches. Moreover, it significantly reduces the issue of congestion-prone traffic with its high-speed adaptive routing strategies. Thus, we can state that the proposed approach overcomes real-world issues like facing delays, sparse reward feedback, and networks of partial observability. In future, we intend to make the model generic and increase the scalability as well. In addition, we need to handle a large, dynamic network where nodes and edges are often in case of mobile ad hoc networks. The model does not support holding any study for mobile ad hoc networks. Moreover, the comparison is done only with the OSPF. It limits to describe the agent's behavior in a dynamic environment as well. The number of nodes considered was 200 only; the results may vary with an increase in the number of nodes.

## References

[1] Chen, B., Zhu, D., Wang, Y., & Zhang, P. (2022). An approach to combine the power of deep reinforcement learning with a graph neural network for routing optimization. Electronics, 11(3), 368.

[2] Munikoti, S., Agarwal, D., Das, L., Halappanavar, M., & Natarajan, B. (2023). Challenges and opportunities in deep reinforcement learning with graph neural networks: A comprehensive review of algorithms and applications. IEEE transactions on neural networks and learning systems.

[3] Tam, P., Ros, S., Song, I., Kang, S., & Kim, S. (2024). A survey of intelligent end-to-end networking solutions: Integrating graph neural networks and deep reinforcement learning approaches. Electronics, 13(5),994.

[4] Zhang, P., Wang, C., Kumar, N., Zhang, W., & Liu, L. (2021). Dynamic virtual network embedding algorithm based on graph convolution neural network and reinforcement learning. IEEE Internet of Things Journal, 9(12), 9389-9398.

[5] Wang, Y., & Liang, X. (2025). Application of Reinforcement Learning Methods Combining Graph Neural Networks and Self-Attention Mechanisms in Supply Chain Route Optimization. Sensors, 25(3), 955.

[6] Liu, W., Zhang, C., Zhou, K., Li, Y., Zhan, F., Xue, W., & Chen, R. (2025). Sequential Decision MARL for Adaptive Traffic Signal Control With Different Intersections Priorities. IEEE Transactions on Intelligent Transportation Systems.

[7] Wang, Y., Wu, H., & Li, R. (2024). Deep graph reinforcement learning for mobile edge computing: Challenges and solutions. IEEE Network.

[8] Sahili, Z. A., & Awad, M. (2023). Spatio-temporal graph neural networks: A survey. arXiv preprint arXiv:2301.10569.

[9] Sharma, A., Sharma, A., Nikashina, P., Gavrilenko, V., Tselykh, A., Bozhenyuk, A., ... & Meshref, H. (2023). A graph neural network (GNN)-based approach for real-time estimation of traffic speed in sustainable smart cities. Sustainability, 15(15), 11893.

[10] Zhong, Y., Sheng, G., Qin, T., Wang, M., Gan, Q., & Wu, C. (2023). Gnnflow: A distributed framework for continuous temporal gnn learning on dynamic graphs. arXiv preprint arXiv:2311.17410.

[11] He, Q., Wang, Y., Wang, X., Xu, W., Li, F., Yang, K., & Ma, L. (2023). Routing optimization with deep reinforcement learning in knowledge defined networking. IEEE Transactions on Mobile Computing, 23(2), 1444-1455.

[12] Chen, T., Yang, L., Wang, Z., & Long, J. (2025). A rule-and query-guided reinforcement learning for extrapolation reasoning in temporal knowledge graphs. Neural Networks, 185, 107186.

[13] Lera, I., & Guerrero, C. (2024). Multi-objective application placement in fog computing using graph neural network-based reinforcement learning. The Journal of Supercomputing, 80(19), 27073-27094.

[14] Almasan, P., Suárez-Varela, J., Rusek, K., Barlet-Ros, P., & Cabellos-Aparicio, A. (2022). Deep reinforcement learning meets graph neural networks: Exploring a routing optimization use case. Computer Communications, 196, 184-194.

[15] Jiang, W., Han, H., Zhang, Y., Wang, J. A., He, M., Gu, W., ... & Cheng, X. (2024). Graph Neural Networks for Routing Optimization: Challenges and Opportunities. Sustainability, 16(21), 9239.

[16] Dey, R. K., & Das, A. K. (2023). Modified term frequency-inverse document frequency based deep hybrid framework for sentiment analysis. Multimedia Tools and Applications, 82(21), 32967-32990.

[17] Jena, S., Brahma, B., Khan, Z., Jyothi, G., Arunachalam, P., & Aggarwal, S. (2025). SBB-Chi2-A2: stacking of bagging-boosting with the blend Chi-square for effective prediction of aortic aneurysm using biomarker profiling. International Journal of Information Technology, 1-8.

[18] Jadon, P., Bhatia, D., & Mishra, D. K. (2024, December). Social Media Text Classification For Hate Speech Detection Using Different Feature Selection Techniques. In 2024 IEEE 4th International Conference on ICT in Business Industry & Government (ICTBIG) (pp. 1-8). IEEE.

[19] Aliyeva, Ç. O., & Yağanoğlu, M. (2025). Deep learning approach to detect cyberbullying on twitter. Multimedia Tools and Applications, 84(19), 20497-20520.

[20] Geng, Z., Xu, J., Wu, R., Zhao, C., Wang, J., Li, Y., & Zhang, C. (2024). STGAFormer: Spatial–temporal gated attention transformer based graph neural network for traffic flow forecasting. Information Fusion, 105, 102228.

[21] Gu, H., Wang, S., Jia, D., Zhang, Y., Luo, Y., Mao, G., ... & Lim, E. G. (2025). Communication Strategy on Macro-and-Micro Traffic State in Cooperative Deep Reinforcement Learning for Regional Traffic Signal Control. IEEE Transactions on Intelligent Transportation Systems.

[22] Raman, R., Kumar, V., Saini, D., Rabadiya, D., Rastogi, S., & Kumbhojkar, N. (2024, September). Energy-Efficient Deep Q-Network for Reinforcement Learning in Wireless IoT Routing Protocols. In 2024 Asian Conference on Intelligent Technologies (ACOIT) (pp. 1-5). IEEE.

[23] Li, W., Song, X., & Tu, Y. (2025). GraphDRL: GNN-based deep reinforcement learning for interactive recommendation with sparse data. Expert Systems with Applications, 273, 126832.

[24] Sharma, A., Sharma, A., & Guo, K. (2025). Intelligent Medical Diagnosis Model Based on Graph Neural Networks for Medical Images. CAAI Transactions on Intelligence Technology.

[25] Sharma, A., Sharma, A., Nikashina, P., Gavrilenko, V., Tselykh, A., Bozhenyuk, A., ... & Meshref, H. (2023). A graph neural network (GNN)-based approach for real-time estimation of traffic speed in sustainable smart cities. Sustainability, 15(15), 11893.