International Journal of Emerging Trends in Computer Science and Information Technology
ISSN: 3050-9246 | https://doi.org/10.63282/3050-9246/ICRTCSIT-126
Eureka Vision Publication | ICRTCSIT'25-Conference Proceeding

Original Article

Using Neural Networks for Packet Routing

Sujay Kanungo
Independent Researcher Boston, USA.

Abstract - Packet Routing has been and is the core of any Networking Node and how it interacts with the neighboring network nodes and it also governs how traffic is routed through these devices. Traditional routing algorithms rely on configured peering rules which govern with which routing nodes each can talk to and which networks it can learn and forward traffic to. This mechanism also protects these nodes from learning rogue networks and forwarding packets to unintended destinations. Some of these approaches suffer from fundamental limitations. Choosing an appropriate data structure is not a straightforward task and often hard to estimate. In this paper I will expand on the idea suggested by the paper A Deep Reinforcement Learning Approach for Adaptive Traffic Routing in Next – gen networks by suggesting next gen algorithms which can expand the concepts of Neural Nets to both Internal and External Routing domains which takes care of machine learning based packet routing in both enterprise and public networks and internet. In this paper we will develop a methodology to use Graph Neural Networks, Deep Reinforcement Learning, Recurrent Neural Networks and Autoencoders and Generative Adversarial Networks to create a system which can learn paths and forward traffic and also maintain the quality of service of those paths.

Keywords - WAN-NN, Neural Networks, Packet Routing

1. Introduction

Packet forwarding stands out as the primary activity in network communications. Incoming packets are examined at the network layer; their destination address is identified; then the packets are set to the next-hop addresses. Consequently, packet forwarding algorithms have served as the foundation for dependable network communications. Past decades have witnessed numerous efforts in the design of packet forwarding algorithms, which are traditionally engineered as procedures, characterized by a sequence of well-defined operations. Possessing fundamental properties such as scalability, efficiency, and accuracy, indeed, they have become heavily embedded in network machinery and systems. Nonetheless, packet forwarding should not be regarded as a one-off mission. Instead, it continues to evolve due to the growth of network infrastructure and the evolution of upper layer applications.

Neural Networks have recently received significant attention in many fields due to their ability to learn and model non-linear and complex relationships. The recent dense traffic in data networks is also making a great effect on the network QoS as well as the performance. The existing packet forwarding does not provide the efficient packet delivery due to the high data traffic as well as inefficient data packet forwarding techniques. There is a necessity to maintain the QoS parameters in order to obtain the high speed and reliable transmission of the packets. Traffic should be analyzed properly in every stage to estimate the status of the traffic in a very simple and effective way. To address these issues, a host-based approach is proposed to integrated NNs with packet forwarding in data networks, the overall data network performance can be improved. The main objective of this integration is to achieve an efficient and reliable packet transmission within a specified time limit.In the rest of the paper, we will discuss the background and motivation, current packet forwarding methods and the issues, integrations of NNs for packet forwarding in packet forwarding along with the concepts of Neural Networks.

2. Background and Motivation

2.1. Traditional Packet Routing Algorithms

Packet-classification algorithms determine the match rule of a data packet that exhibits the longest prefix match with the packet's destination IP address. Classification takes place over multiple dimensions simultaneously, including packets source and destination IP addresses, source and destination port numbers, protocol number, and IP transit time to destination. Traditional packet-forwarding algorithms are based on decision trees, tuples, exhaustive search and approximately homomorphic encryption. Decision-tree packet classification starts with a root node containing all rules. Each internal node partitions the rule set according to one or more network-field cuts. The partitioned rules are pushed down to sub-nodes, and the algorithm recurses over the full classification tree until the size of remaining rules exceeds the number of packet's header bits. Once the rule set is expandable, the remainder of each rule is transformed into a tuple and put into the leaf node for packet matching. Short prefixes are extended to full length, and prefix, port, and range values are mapped into the interval $[0, M-1]$, where M is a sufficiently large number. Several decision-tree algorithms produce a packet-forwarding system:

HiCuts packet classification uses cuts in only one dimension, HyperCuts uses cuts in multiple dimensions, HyperSplit combines rule-based space decomposition with optimized recursion, EffiCuts reduces rule replication and imbalance rate by

heuristics such as separable trees and node co-location, and CutSplit integrates equal-sized and equal-dense cuts. Alternatives to decision trees include tuple space search, RFC, and DCFL algorithms, but these methods are less popular because of slower-paced execution or larger memory space requirements. Hardware-assisted packet classification exploits TCAMs, GPUs, and FPGAs. Most existing packet-forwarding algorithms are based on heuristics and algorithms, so the performance of the algorithms is limited. Although existing heuristic-based methods offer practical solutions for packet classification and routing, they could benefit from a learning-based or analytical framework that generates efficient, scalable trees. Some routing protocols currently prevalent in the industry are OSPF And BGP Open Shortest Path First (OSPF) is a link-state interior gateway protocol (IGP) used within a single autonomous system to determine the most efficient routes for data packets. Each router in an OSPF network maintains a map of the network topology by exchanging Link State Advertisements (LSAs) with its neighbors. Using this shared database, every router independently computes the shortest path to all destinations using Dijkstra's Shortest Path First (SPF) algorithm. OSPF supports features like hierarchical area division, load balancing, and fast convergence, making it suitable for large enterprise and service provider networks.

Border Gateway Protocol (BGP), on the other hand, is an exterior gateway protocol (EGP) that governs how data is routed between autonomous systems (ASes) across the global Internet. BGP operates as a path-vector protocol, where routers exchange routing information containing the list of ASes (AS-path) that data must traverse to reach a destination. Route selection in BGP is based on policies rather than shortest paths, considering factors like AS-path length, local preferences, and multi-exit discriminators. Unlike OSPF, which prioritizes efficiency and speed within an organization, BGP focuses on policy control, scalability, and stability between organizations, forming the backbone of inter-domain Internet routing.

2.2. Limitations of Conventional Methods

Conventional packet forwarding methods rely on network nodes maintaining and traversing data structures such as trees or lists. Since the data structures follow a predefined form, a fixed procedure is sufficient to determine the output port for each incoming packet. This simplicity enables the achievement of both high forwarding speed and accuracy. Nevertheless, these approaches suffer from fundamental limitations. Choosing an appropriate data structure is not a straightforward task and often relies on heuristics. Because heuristics are inherently imperfect, the resulting forwarding performance and data structure size have bounds that are often hard to estimate. Furthermore, once a node is locked into a particular data structure form, it cannot flexibly adapt to channel conditions or network dynamics. To surmount these restrictions, a more effective and adaptable method is indispensable. Below are the limitations of some of the current known Packet routing algorithms.

Open Shortest Path First (OSPF), while highly efficient for intra-domain routing, has several limitations. One of the primary challenges is scalability — as the network grows, the number of routers and links increases, expanding the size of the link-state database each router must maintain. This can lead to higher CPU and memory consumption, especially during topology changes when all routers must recalculate their routing tables using Dijkstra's algorithm. Additionally, OSPF requires careful area design and configuration to remain efficient. Poorly planned hierarchies or area boundaries can result in routing inefficiencies and increased convergence times. Furthermore, OSPF's convergence process, although faster than distance-vector protocols, can still be slow in very large or unstable networks due to frequent flooding of Link State Advertisements (LSAs).

Border Gateway Protocol (BGP), on the other hand, faces limitations primarily due to its policy-based and decentralized nature. BGP prioritizes stability and administrative control over speed, which leads to slow convergence after network changes or failures—potentially lasting minutes. Because BGP operates on the global Internet, scalability is another concern, as it must handle routing information for hundreds of thousands of networks (prefixes). The lack of strong built-in security mechanisms also makes BGP vulnerable to issues such as prefix hijacking, route leaks, and misconfigurations, which can disrupt large portions of the Internet. Additionally, BGP's policy complexity can make configuration and troubleshooting difficult, requiring careful tuning to balance performance, security, and business relationships between autonomous systems.

3. Neural Networks

3.1. What are Neural Networks

Artificial Intelligence (AI) encompasses algorithms supporting machine perception, reasoning, and actuation; such algorithms can be **classified** into symbolic and sub-symbolic approaches. Sub-symbolic approaches model real-world data in the form of feature vectors; thus, they are generally suited for physical problems but less appropriate for traditional computer science problems (e.g., computer vision versus software debugging). Sub-symbolic techniques include k-nearest neighbors, support vector machines, Bayesian networks, and neural networks, the latter being the most prominent technique comprising a highly parallelized architecture inspired by the animal brain.

3.2. Types of Neural Networks

Neural networks form the cornerstone of contemporary artificial intelligence applications. They are sophisticated programs that model data by emulating the architecture and operational principles of biological neural systems. A neural network comprises numerous interconnected nodes arranged in layers that process input data to produce specific outputs. The two primary categories include convolutional neural networks (CNNs) and graph neural networks (GNNs). CNNs excel when inputs exhibit a grid-like structure—for instance, pixels in images—by capturing spatial hierarchies and patterns. GNNs, conversely, are tailored to graph-structured data, where relationships and interactions among entities are paramount. Both CNNs and GNNs have demonstrated considerable promise in routing tasks due to their structural affinity for interpreting network data.

3.3. Neural Net Architecture

Neural networks map arbitrary input spaces to response spaces through composite hidden layers linked by adaptive synaptic weights [2]. Individual neural units, or perceptron, form the building blocks of these hierarchical structures, each computing the product of inputs and corresponding weight coefficients before aggregation.A typical multilayer architecture incorporates hidden layers operating in a feed-forward manner, utilizing the preceding layer's output as input. The final output is derived based on a key neuron that aggregates the activities of the preceding layer. Unlike biological counterparts, these architectures feature industrial units without inherent memory, focusing solely on current synaptic weights and inputs. This design renders the output at any time dependent exclusively on the present input pattern, enabling efficient encoding of arbitrary nonlinear functions across multiple scales.

4. Integrating Neural Networks in Networking

Integrating neural networks into packet forwarding software calls for selecting suitable architectures based on the relative priorities of fast adaptation, very low latency querying, scalability, and advanced generalization. Training methodologies can leverage convolutional approaches to graph data, supervised learning with historical records of successful routes and known unreachable destinations, or reinforcement learning with an embedded simulator replicating dynamic topology phases.A. Challenges and short comings of current research related to Neural Networks for Packet Routing Let's discuss the implementation focused shortcomings of the paper "A deep Reinforcement Learning Approach for Adaptive Traffic Routing in Next-gen Networks" and similar papers like that.

- Action space is too narrow – The agents only choose among precomputed shortest paths, which rule out common traffic engineering actions like splitting traffic across multiple paths, changing link weights, or pushing segment/MPLS steering. That severely limits optimality and makes results hard to compare to Wide area Network Traffic engineering baselines that allow fractional flow allocations
- Reward omits key service metrics and operational cost- the reward combines normalized throughput and delay per flow, but ignores packet loss, jitter/tail latency, SLA classes, policy violations, reconfiguration churn, and routing stability- all first-class constraints in production networks. This can produce routes that look "good" in the simulator but violate SLOs in practice.
- State misses critical link-level observability – the "link-state" typically a concept used in OSPF is essentially an adjacency matrix; the rich feature is node-level (degree, centrality, ingress/egress, etc.) There's no explicit per-link utilization, queue depth, ECN marks, capacity, propagation/queuing delay, SRLG info- signals operators depend on for traffic engineering. That weakens the agent's ability to avoid hot links to meet latency budgets.
- Centralized, per-flow path picking does not address control plane/data – plane realities. The design assumes an SDN Controller can feed fresh telemetry and push per flow decisions in real time. The papers do not model telemetry staleness, controller latency, roll-out safety (two-phases), drain timers, or hardware TCAM limits for flow rules. In practice that will dominate if an whether an RL policy is deployable.
- Limited baselines and external validation – Results compare mainly against OSPF rather than standard WAN TE optimizers
- No treatment of failures and fast-reroute- WANs demand sub-50ms local protection and robust behavior under link/node/plane failures; the paper does not evaluate convergence or protection behaviors under failures, which is mandatory for routing protocols.
- Scalability unclear- The evaluation mentions only two topologies with synthetic traffic, trained in Gym. There's no complexity analysis for hundreds of nodes/ links, no evidence of zero-shot generalization to unseen topologies and no ablating how DQN scales as the number of candidate paths explodes.
- Safety and constraints not enforced – There's no mechanism for hard constraints such as setting Quality of Service constraints.

4.1. What the current research does not cover?

- Traffic Engineering formulation – Wide Area Networking Traffic Engineering is usually a multi-commodity flow problem with demand matrices, flow splitting and objective trade-offs. The paper's per flow path selection doesn't model this; state-of-the-art WAN work explicitly optimizes global allocations and shows near optimality

- Protocol/feature surface (MPLS/BGP policy) Production WANs rely on BGP Policies, MPLS /SR-TE, class-based queues, link affinities/constraints, and ECMP. The paper does not cover these.

5. Neural network architecture for packet forwarding

Broad Classes of Neural Network Architectures. Traditional packet-forwarding algorithms determine the next-hop for each incoming packet according to a few fields of the packet headers. To replace such algorithms, a neural network can also output the appropriate next-hop port when receiving those truncated packet-header fields as an input. Different packet-forwarding algorithms make decisions for different reasons, and different neural-network architectures are appropriate for different schemes. For instance, the decision process of the simple destination-based routing algorithm is similar to that of feed-forward neural networks or Long Short-Term Memory (LSTM) networks (Simple Recurrent Networks or SRNs). Both kinds of models perform inferences based solely on the data that are currently available without requiring additional information. An example is the learned Smallest-Prefix-Match (SPM) algorithm, which addresses the largest-prefix-matching problem with a relatively compact feed-forward network. Convolutional neural networks (CNNs) and graph neural networks (GNNs) find a place in network systems as well. Routing algorithms attempt to forward packets according to the entire path in a supply network; specifying such a path requires knowing more than just the packet header. Specifically, path computations require information about the overall network topology and routing table. CNNs and GNNs are a natural fit for representing interconnected systems, such as road networks, social networks and distributed networks, and therefore can learn path-finding behaviors to assist routing decisions.

5.1. How to replace BGP using Neural Nets

The Border Gateway Protocol (BGP) is the current de facto inter-domain routing protocol for the Internet. While BGP has enabled global connectivity, it has limitations in terms of convergence time, security, and manual policy management. Neural networks, particularly when combined with reinforcement learning, offer a potential path toward replacing or augmenting BGP in the future.

5.1.1. Routing Decision Making

Graph Neural Networks (GNNs): By modeling the Internet as a graph of Autonomous Systems (ASes), GNNs can learn optimal path selection strategies beyond the static path-vector logic of BGP.

5.1.2. Policy and Traffic Engineering

Deep Reinforcement Learning (DRL): Neural agents can dynamically balance traffic loads, minimize latency, and optimize operational costs, effectively replacing manual policy configurations in BGP.

5.1.3. Failure Recovery and Convergence

Recurrent Neural Networks (RNNs, LSTMs, GRUs): By analyzing historical routing data, these models could predict link failures and proactively compute backup routes, offering faster convergence than traditional BGP.

5.1.4. Security and Anomaly Defense

Autoencoders and Generative Adversarial Networks (GANs): Neural networks could detect prefix hijacks, leaks, or coordinated attacks in real-time and enforce corrective routing decisions.

5.2. Designing Neural Network based routing for both Interior Gateway and Exterior Gateway

we will call this Algorithm WAN-NN

5.2.1. High-level architecture (single embedded router node)

Graph Encoder (GNN) — converts local topology + neighbor embeddings → node embedding.

- Input: local node features, neighbor IDs, link features.
- Output: node embedding vector hvh_vhv.

Policy / Decision Head (RL / DNN) — takes node embedding + destination embedding → action distribution (next-hop or route).

- Trained with RL (PPO/DQN) plus supervised signals from control-plane/telemetry.

Temporal Predictor (RNN: LSTM/GRU) — predicts link failure probability / metric drift for next T seconds. Used to proactively precompute alternate paths.

Security Module (Autoencoder / Discriminator) — anomaly detection on BGP/LSA updates; flags suspicious updates and optionally blocks.

Co-learning Client — performs local training, compresses gradients/weights, and exchanges updates with peers using a secure, bandwidth-efficient protocol (federated averaging or gossip).

Control-plane Shim — lightweight wrapper that: a) obtains local topology / link-state, b) applies model's decisions to forwarding plane, c) falls back to OSPF/BGP if model is uncertain.

5.3. Algorithm / flow (per-router)

Initialization

- Preload a small GNN + RL policy + LSTM + Autoencoder (weights can be randomized or seeded from an initial global model).
- Initialize trust & neighbor list (peers for co-learning).

Continuous loop (real-time inference):

5.3.1. Observe: collect local state S = {neighbor list, link metrics, local prefix table, traffic matrix samples, recent LSAs/BGP updates}.

5.3.2. Encode: compute node embedding hv=GNN(S)h_v = \mathrm{GNN}(S)hv=GNN(S).

Predict:

- next_hop_probs = PolicyHead(h_v, dest_embedding)
- failure_probs = LSTM(history)
- anomaly_score = Autoencoder(S_updates)

Decide: choose action using policy. If anomaly_score > threshold → trigger defensive policy (isolate route / consult peer).

Apply: program forwarding plane or inject short-lived routing preference to local control plane.

Log experience tuple(s) for training: (state, action, reward, next_state, done).

- Reward = composite metric (latency reduction, packet loss improvement, policy compliance penalty).

Periodic local training (every N seconds or when sufficient data):

- Sample local minibatch from experience buffer.
- Update:
  - RL policy via PPO / actor-critic on local rewards.
  - GNN via supervised or RL signal (backprop through decision head).
  - LSTM via next-step prediction loss.
  - Autoencoder via reconstruction loss.
- Produce compressed model update (delta weights or gradients).

5.4. Co-learning (distributed) protocol

Two practical options (pick depending on network dynamics / bandwidth):

Option 1 — Federated Averaging (periodic, lightweight)

- Each node does local SGD for several steps → compute weight delta Δw\Delta wΔw → compress → securely send to neighbors / an aggregator (if available).
- Aggregation: either a local leader or chained averaging (each node averages with neighbors) → new local model = average of received + local.

Option 2 — Gossip / Decentralized Averaging (asynchronous, robust)

- Nodes periodically pick one or more neighbor peers and exchange compressed model updates / embeddings.
- On receiving an update, perform weighted average: wnew=αwlocal+(1−α)wpeerw_{new} = \alpha w_{local} + (1-\alpha) w_{peer}wnew=αwlocal+(1−α)wpeer.
- Repeat until weights converge.

Security & privacy

- Use Secure Aggregation (additive masking) or homomorphic-friendly lightweight crypto if privacy needed.
- Optionally apply Differential Privacy (add calibrated noise) to updates to prevent model inversion.

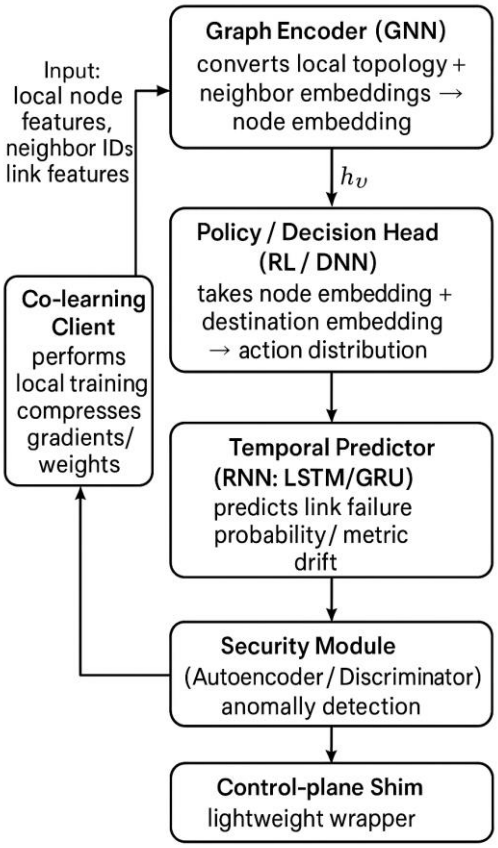

Figure 1. GNN-Based_Network_Agent_with_Temporal_Security_Modules

5.5. Example Scenarios with WAN-NN

5.5.1. Scenario Overview

- **Autonomous System (AS):** AS65001
- **Goal:** Learn the best route to reach prefix 192.0.2.0/24 (destination).
- **Routers:** R1, R2, R3, R4

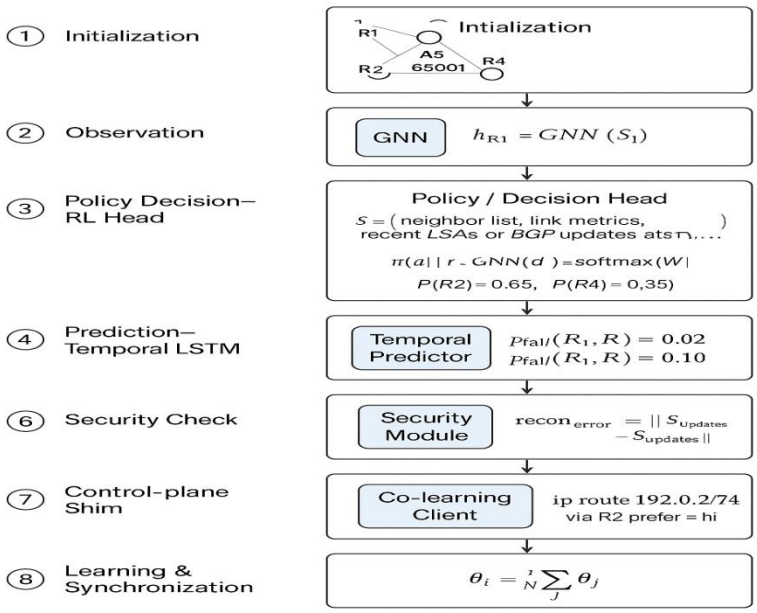

Figure 2. Graph Neural Network–Based Reinforcement Learning Framework for Secure and Adaptive Routing

Links:

- R1–R2 (10 Gbps, latency 5 ms)
- R2–R3 (5 Gbps, latency 15 ms)
- R3–R4 (1 Gbps, latency 25 ms)
- R1–R4 (direct, 2 Gbps, latency 30 ms)

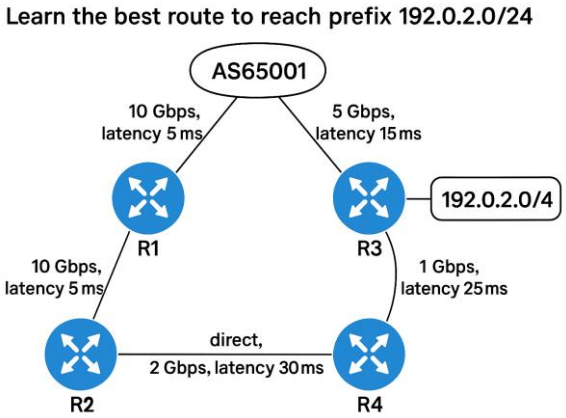

Figure 3. Topology and Link Metrics for Route Optimization

5.6. Origin AS (customer / originator): AS65003 *— announces 192.0.2.0/24 (prefix owner).*

- **Transit / peers**:
- **AS65001** — contains routers **R1, R2**.
- **AS65002** — contains routers **R3, R4**.
- **Inter-AS links**:
- R2 (AS65001) — R3 (AS65002) (peer link)
- R4 (AS65002) — R5 (AS65003) (customer link into origin)
- R1 and R2 internal in AS65001; R3 and R4 internal in AS65002.
- **Goal**: routers across AS65001 and AS65002 cooperatively learn and optimize routes to 192.0.2.0/24 while respecting export filters, RPKI validation, and privacy constraints.

How the learning will work here BGP is just used for flow transmission and not for actual learning it is envisioned that in the intial stages of the development of these methods there will be time when BGP and OSPF will co-exist with these WAN-NNs.

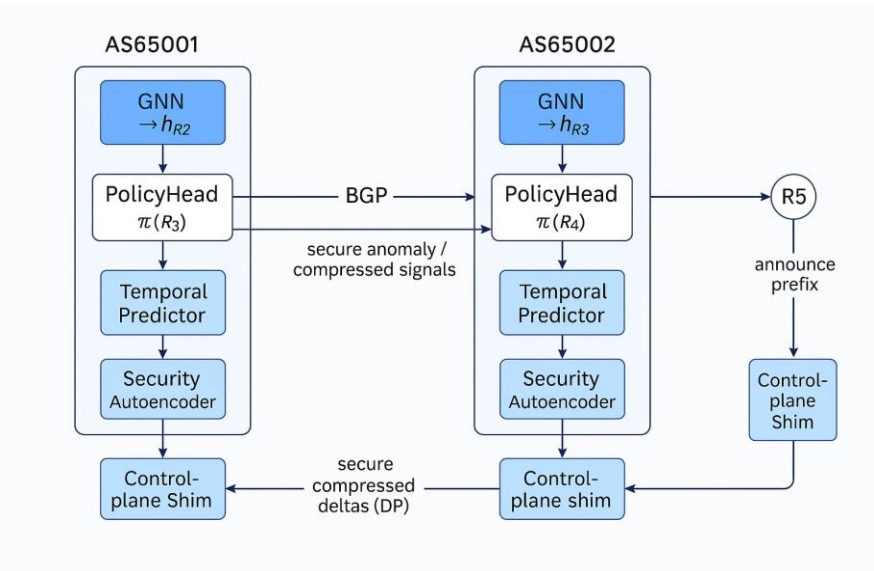

Figure 4. Federated GNN-RL Architecture for Secure BGP Policy Exchange

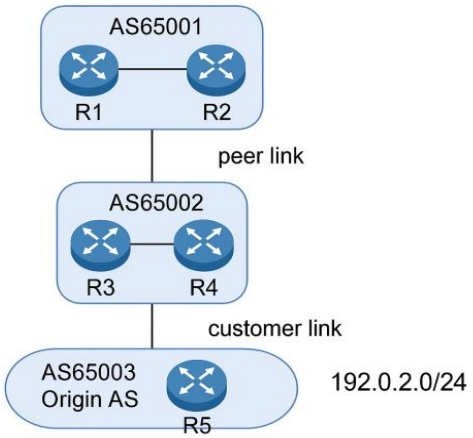

Figure 5. Inter-AS Topology with Peer and Customer Links

6. Conclusion

The integration of neural networks into packet routing represents a transformative step toward building self-optimizing and adaptive communication networks. Traditional routing protocols such as OSPF and BGP have provided decades of stability but remain limited by static configurations, slow convergence, and scalability constraints. The proposed WAN-NN architecture demonstrates that by combining Graph Neural Networks (GNNs), Deep Reinforcement Learning (DRL), and recurrent models such as LSTMs, routing decisions can evolve dynamically in response to real-time network telemetry and topological changes. Furthermore, incorporating autoencoders and generative adversarial networks (GANs) introduces an additional layer of intelligence for anomaly detection, fault prediction, and adaptive path optimization.Through federated co-learning and decentralized model synchronization, WAN-NN allows routers to collaborate without exposing private routing data, preserving both scalability and security. This distributed intelligence enables faster convergence, proactive failure recovery, and policy-aware optimization beyond what is feasible with traditional routing systems. While practical deployment challenges remain—such as ensuring interoperability, enforcing QoS guarantees, and handling model drift—the framework outlined in this paper establishes a concrete foundation for AI-driven routing systems in next-generation networks. Future research will focus on extending WAN-NN's adaptability to large-scale, heterogeneous environments and integrating trust-aware and explainable AI models to make neural routing both transparent and verifiable.

References

[1] E. Liang, H. Zhu, X. Jin, and I. Stoica, "Neural Packet Classification," 2019. [PDF]

[2] X. You, X. Li, Y. Xu, H. Feng et al., "Toward Packet Routing with Fully-distributed Multi-agent Deep Reinforcement Learning," 2019. [PDF]

[3] A. Abrol, P. Murali Mohan, and T. Truong-Huu, "A Deep Reinforcement Learning Approach for Adaptive Traffic Routing in Next-gen Networks," 2024. [PDF]

[4] T. Valerrian Pasca S, S. Sairam Prasad, and K. Kataoka, "AMPF: Application-aware Multipath Packet Forwarding using Machine Learning and SDN," 2016. [PDF]

[5] C. Cascone, R. Bifulco, S. Pontarelli, and A. Capone, "Relaxing state-access constraints in stateful programmable data planes," 2018. [PDF]

[6] P. Cui, H. Pan, Z. Li, J. Wu et al., "NetFC: enabling accurate floating-point arithmetic on programmable switches," 2021. [PDF]

[7] L. McHale, P. V Gratz, and A. Sprintson, "Flow Correlator: A Flow Table Cache Management Strategy," 2023. [PDF]

[8] Arpit Garg, S Rautaray, Devrajavans Tayagi. Artificial Intelligence in Telecommunications: Applications, Risks, and Governance in the 5G and Beyond Era. International Journal of Computer Techniques – Volume10Issue1,January - February – 2023. 1-19.

[9] Varinder Kumar Sharma - Federated Learning in Mobile and Edge Environments for Telecom Use Cases - International Journal of Innovative Research and Creative Technology (www.ijirct.org) Volume 10 Issue 1 January-2024.DOI: https://doi.org/10.5281/zenodo.17062956

[10] Amrish Solanki1, ShrikaaJadiga, AI Applications for Improving Transportation and Logistics Operations, International Journal ofINTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING, International Journal of Intelligent Systems and Applications in EngineeringIJISAE, 2024, 12(3), 2607–2617.