Graph-Based Learning for Intelligent Network Management: Adaptive Restoration, Reliability Prediction, and Traffic Forecasting

by

Isaac Ampratwum

Thesis submitted to the University of Ottawa

in partial Fulfillment of the requirements for the

Doctor of Philosophy in

Electrical and Computer Engineering

School of Electrical Engineering and Computer Science

Faculty of Engineering

University of Ottawa

Abstract

Modern communication networks are becoming increasingly complex due to rising traffic demands, stringent reliability requirements, and the need for adaptive management across optical and wireless infrastructures. Traditional rule-based and static management approaches are often inadequate for handling the structural, temporal, and operational complexity of these environments. This thesis investigates Graph Neural Networks (GNNs) as a unifying learning paradigm for intelligent network management, and demonstrates how task-specific integration with reinforcement learning, temporal sequence models, and evolutionary optimization can improve decision-making across diverse networking problems.

Three representative network management challenges are addressed. First, a GNN-enhanced deep reinforcement learning framework is developed for dynamic restoration in Wavelength Division Multiplexing (WDM) networks. By encoding topology and failure states as graph-structured inputs, the framework enables topology-aware routing and wavelength assignment, achieving faster and more effective recovery than conventional heuristic restoration methods. Second, a graph-based predictive framework is proposed for proactive radio link failure prediction in 5G networks. By combining spatial relationships among neighboring sites with temporal and environmental context, the model improves the early detection of service disruptions and supports proactive reliability management. Third, a spatio-temporal GNN-based traffic forecasting model, augmented with Transformer learning and Genetic Algorithm-based hyperparameter optimization, is introduced for 6G network environments. This model captures both structural and temporal traffic dependencies, improving forecasting accuracy and enabling more effective predictive resource allocation.

Taken together, these contributions show that graph-based learning provides a co-

herent and effective foundation for intelligent network management across resilience, reliability, and forecasting tasks. Rather than proposing a single universal architecture for all network problems, this thesis demonstrates how a common graph-centered perspective can be adapted to different management objectives through appropriate learning and optimization mechanisms. The results establish GNN-based modeling as a practical pathway toward more adaptive, predictive, and autonomous communication networks.

Acknowledgements

I would like to begin by expressing my utmost gratitude to the Almighty God for His direction and guidance throughout my academic journey. I extend my heartfelt thanks to my supervisor, Professor Amiya Nayak, for his invaluable counsel and support during my thesis work. Finally, I wish to express my profound gratitude to my amazing wife, my kids and friends for their unwavering support and inspiration as I conducted my research and wrote my thesis. Without them, this feat would not have been possible. I am thankful.

Contents

List of Tables

List of Figures

Acronyms

5G Fifth-Generation.

6G Sixth-Generation.

AI Artificial Intelligence.

API Application Programming Interface.

APIs Application Programming Interfaces.

AR/VR Augmented Reality / Virtual Reality.

AUC Area Under the Curve.

BBE Background Block Errors.

BNN Bayesian Neural Network.

BSS Business Support Systems.

CI/CD Continuous Integration / Continuous Deployment.

CNN Convolutional Neural Network.

DBGRN Dynamic Bernstein Graph Recurrent Network.

DBPF Dynamic Bernstein Polynomial Filtering.

DFT Discrete Fourier Transform.

DRL Deep Reinforcement Learning.

EON Elastic Optical Network.

FL Federated Learning.

GA Genetic Algorithm.

GAT Graph Attention Network.

GCN Graph Convolutional Network.

GNN Graph Neural Network.

GPU Graphics Processing Unit.

GRU Gated Recurrent Unit.

ILP Integer Linear Programming.

Informer Efficient Transformer for Long Sequence Time-Series Forecasting.

IoT Internet of Things.

KPI Key Performance Indicator.

LR Logistic Regression.

LSTM Long Short-Term Memory.

ML Machine Learning.

MLOps Machine Learning Operations.

MSTNN Multi-Step Temporal Neural Network.

NMS Network Management System.

NOC Network Operations Center.

OADM Optical Add-Drop Multiplexer.

OSS Operations Support Systems.

OXC Optical Cross-Connect.

PR–AUC Area Under the Precision–Recall Curve.

QoS Quality of Service.

RAN Radio Access Network.

RL Reinforcement Learning.

RLF Radio Link Failure.

RNN Recurrent Neural Network.

ROC–AUC Area Under the Receiver Operating Characteristic Curve.

RSA Routing and Spectrum Assignment.

SDGC Spectral Dynamic Graph Construction.

SDN Software Defined Networking.

SGCRN Spatio-Temporal Graph Convolutional Recurrent Network.

SLO Service Level Objective.

SoTA State of the Art.

SVM Support Vector Machine.

Transformer Transformer Neural Network Architecture.

TSGAN Time-Series Graph Attention Network.

URLLC Ultra-Reliable Low-Latency Communications.

WDM Wavelength Division Multiplexing.

Chapter 1

Introduction

1.1 Overview

1.1.1 Intelligent Network Management in the 5G/6G Era

The rapid evolution of communication networks toward fifth and sixth generations (5G and 6G) has brought unprecedented growth in traffic volumes and performance expectations. Global IP and mobile data traffic continue to surge, with forecasts predicting mobile data to reach 282 exabytes per month by 2027 [1]. This explosion is fueled by data-intensive applications like ultra-high-definition video, augmented/virtual reality (AR/VR), and massive Internet of Things (IoT) deployments. Next-generation use cases such as Industry 4.0 automation, autonomous driving and telemedicine demand not only higher capacity but also stringent latency and reliability. For instance, 6G networks are envisioned to deliver terabits-per-second throughput, microsecond-level latencies, and near-perfect reliability [2], an order-of-magnitude improvement over 5G's targets for ultra-reliable low-latency communications (URLLC). In parallel, optical fiber networks (e.g. wavelength-division multiplexing, WDM) form the backbone that inter-

connects data centers and cellular infrastructure, carrying these massive traffic loads. Optical WDM systems offer terabit-scale capacity, but a single fiber cut or equipment failure can disrupt enormous volumes of data. Minimizing downtime is paramount, as network outages cost enterprises a lot of money. In this context, intelligent network management, the ability to dynamically monitor, control, and optimize network behavior has become a critical enabler to meet 5G/6G era requirements for capacity, latency, and reliability. Meeting these ambitious targets is exceptionally challenging with conventional network management approaches. Modern 5G radio access networks (RANs) are far denser and more complex than previous generations, integrating high-frequency millimeter-wave cells that are sensitive to environmental factors like rain or obstruction [3]. 6G is poised to introduce even more heterogeneity including terahertz links, aerial base stations and intelligent surfaces, further complicating real-time decision-making. Meanwhile, user traffic patterns are becoming more unpredictable due to on-demand services and mobility, stressing the need for adaptive control. Clearly, novel approaches rooted in machine intelligence are required to manage network resources, failures, and traffic in a proactive and optimal manner. This is the driving motivation for our work: leveraging advanced learning algorithms to enable autonomous network management, wherein networks can self-optimize and self-heal to satisfy the formidable performance demands of the 5G/6G era.

1.1.2 Limitations of Traditional Approaches

Traditional network management solutions struggle to cope with the scale and dynamics of today's networks. Legacy methods for network restoration, failure management, and traffic engineering were largely designed for simpler scenarios and often rely on static configurations or human-defined heuristics. In the context of modern high-speed networks,

these approaches exhibit several key limitations:

- **Restoration in Optical Networks**: Conventional WDM optical network restoration typically employs pre-designed protection schemes or basic dynamic rerouting protocols. While simple, these methods are inflexible and resource-inefficient. Dynamic restoration algorithms exist but have received comparatively less attention and often require longer recovery times. Traditional restoration schemes are unable to fully exploit the network's capacity or rapidly adapt to complex failure scenarios in large-scale optical meshes.

- **Failure Prediction and Resilience**: Network failure management in current cellular systems (4G/5G) is largely reactive – e.g. a radio link failure (RLF) is addressed after it occurs through handover or restart procedures. Conventional RLF mitigation relies on threshold-based triggers such as signal quality or packet loss exceeding a limit, which often cannot give advance warning. Environmental factors like precipitation, temperature, wind that degrade 5G mmWave links [3] are usually ignored by traditional models, leading to unexpected outages. Historical approaches might use simple regression on performance counters, but spatial correlations (e.g. a storm affecting multiple sites) and temporal patterns are not effectively captured[3]. Thus, operators lack accurate prediction of failures before they happen, resulting in reduced reliability and higher latency due to unanticipated link drops. The inability of traditional methods to integrate multi-modal data such as weather forecasts or cross-cell interference metrics and to learn complex failure indicators hampers timely preventative actions in modern RANs.

- **Traffic forecasting and Resource Allocation**: Classical traffic engineering approaches plan capacity using coarse forecasts (monthly/yearly trends) or simple time-series models (e.g. ARIMA). These statistical models are computationally

efficient but assume linearity and stationarity, making them inadequate for the volatile, bursty traffic patterns in contemporary networks [4]. They struggle to capture complex spatio-temporal dependencies like how traffic loads at one node may impact others, or how periodic patterns mix with anomalous events. As a result, traditional forecasting can be significantly inaccurate, leading to either over-provisioning or under-provisioning.

These gaps highlight a pressing need for more intelligent, learning-driven approaches to network management. Restoration algorithms must become situation aware and adaptive; failure management should shift from reactive to proactive via prediction; and traffic engineering should leverage rich data to anticipate demand. The following section reviews emerging techniques that promise to overcome these limitations.

1.1.3 Graph Neural Networks (GNNs)

GNNs are a class of deep learning models tailored for graph-structured data. Unlike traditional neural networks that assume independent inputs, GNNs can learn representations of nodes or edges while accounting for the relations and connectivity in graphs. Communication networks naturally form graphs where nodes can represent routers, switches, base stations, etc., and links represent physical or logical connections. By applying neural message-passing and aggregation over the graph, GNNs can capture complex topological patterns and spatial correlations. Recent advances in GNN models such as Graph Convolutional Networks, Graph Attention Networks, etc have been leveraged in various networking problems [5]. For example, GNN-based models have achieved success in routing optimization, learning efficient paths in large network topologies by generalizing across different network layouts [5]. GNNs have also been used for network performance modeling e.g. predicting throughput or delays by effectively learning the impact of

topology on flow outcomes [6]. Importantly, GNNs offer scalability and generalization: a well-trained GNN can potentially apply to networks larger or different from those seen in training, as it learns the underlying relational structure. This makes GNNs extremely attractive for intelligent network management, where the goal is to interpret and act on global network state (a graph) rather than isolated data points.

1.1.4 Deep Reinforcement Learning (DRL)

DRL combines reinforcement learning, where an agent learns to make decisions through trial-and-error interactions with the environment with deep neural networks as function approximators. In networking, DRL enables an autonomous agent or controller to learn optimal or near-optimal policies for complex control tasks, guided by a reward signal such as throughput maximization or latency minimization. Recent advances in DRL including DQN, policy gradients, actor-critic methods have made it feasible to train agents even in high-dimensional, dynamic environments.

DRL has been applied to a wide array of network management problems: from routing and traffic engineering (learning dynamic routing policies that outperform static algorithms) to resource allocation (allocating spectrum, power, or slices based on current demand) [5]. A key strength of DRL is its ability to handle uncertainty and adapt: the agent continuously improves its decisions as it experiences new network states, which is ideal for non-stationary scenarios like time-varying traffic or unforeseen failures.

1.1.5 Genetic Algorithms (GAs)

GAs are a classic evolutionary optimization technique inspired by natural selection. They maintain a population of candidate solutions and iteratively evolve them through selection, crossover, and mutation operators. GAs have a long history in network optimization

tasks [7].

They have been used to tackle problems such as optimal network topology design, routing, frequency assignment, and others, especially where the search space is huge and discrete making gradient-based optimization infeasible. One appeal of GAs is their generality and flexibility: they make few assumptions about the underlying problem and can escape local optima by exploring diverse solutions. Although evolutionary algorithms fell out of favor for a time due to the rise of more efficient convex optimization and machine learning methods, they are seeing a resurgence for specific tasks like hyperparameter optimization and multi-objective optimization. In the context of intelligent networking, GAs can complement learning-based models by finding good configurations that might be hard to discover via gradient descent alone. For instance, a GA can tune the hyperparameters of a deep learning model such as a GNN or a Transformer to maximize prediction accuracy or reward, by efficiently searching the hyperparameter space. This evolutionary learning approach is beneficial when integrating new AI models into network management, as it can automate the model selection and parameter tuning process.

1.2 Thesis Statement and Scope

This thesis argues that Graph Neural Networks (GNNs) provide an effective unifying foundation for intelligent network management because communication networks are inherently relational systems whose states, dependencies, and constraints are naturally expressed as graphs. Across optical and wireless domains, key management functions such as restoration, reliability prediction, and traffic forecasting all require models that can capture structural dependencies between network elements while adapting to dynamic operating conditions. As outlined in this thesis, traditional methods often struggle with this combination of structural complexity, uncertainty, and scale, whereas graph-based

learning offers a principled way to represent and reason over network state.

The central claim of this dissertation is not that a single universal model can solve all network management problems. Rather, the claim is that a graph-centered learning perspective can be systematically adapted across multiple management tasks, with task-specific learning or optimization mechanisms layered on top as needed. In this thesis, GNNs serve as the common representational backbone, while complementary methods are introduced according to the demands of each problem: deep reinforcement learning for sequential decision-making in restoration, temporal sequence modeling for failure prediction and traffic forecasting, and genetic algorithm-based optimization for model configuration in forecasting. This thesis therefore focuses on the role of graph-based intelligence as the unifying paradigm, not on presenting DRL, Transformers, and GA as equally central methodological pillars.

The scope of the dissertation is defined through three representative problems in intelligent network management:

- dynamic restoration in optical WDM networks
- proactive radio link failure in 5G networks and
- spatio-temporal traffic forecasting in 6G-oriented network environments

These three problems are intentionally chosen because together they span three complementary operational objectives:

- resilience, through restoration after disruption,
- reliability, through prediction of impending failure, and
- proactive planning, through forecasting of future demand.

Taken together, they provide a coherent demonstration of how graph-based learning can support reactive, predictive, and anticipatory network management across heterogeneous communication systems.

This thesis does not aim to provide a complete end-to-end autonomous networking stack, nor does it claim that all intelligent networking tasks should be solved using the same architecture. Instead, its contribution is to establish and validate a common graph-based perspective for network management, and to show how this perspective can be tailored to the distinct structural, temporal, and optimization requirements of different network control problems.

1.3 Research Objective

The objective of this thesis is to investigate how graph-based learning can enable more adaptive, scalable, and effective network management in communication systems characterized by structural complexity, temporal dynamics, and operational uncertainty. Specifically, the thesis explores how GNNs can be used as a common modeling framework across three important classes of management problems: restoration, failure prediction, and traffic forecasting.

To achieve this objective, the thesis develops three task-specific frameworks. In optical WDM restoration, GNN-based state representation is combined with deep reinforcement learning to support topology-aware sequential decision-making under failure. In 5G radio link failure prediction, graph-based spatial modeling is combined with temporal learning to improve proactive reliability management. In 6G traffic forecasting, GNN-based spatial representation is combined with temporal sequence modeling and genetic algorithm-based hyperparameter optimization to improve predictive resource planning. Through these studies, the thesis evaluates both the generality of graph-based represen-

tations and the value of combining them with complementary learning and optimization techniques suited to each task.

The broader aim is to demonstrate that graph-centered learning can bridge the gap between static, heuristic network management approaches and more adaptive, predictive, and operationally relevant AI-driven solutions for future communication networks.

1.4 Dissertation-Level Contributions

This thesis makes the following dissertation-level contributions to intelligent network management:

- **A unifying graph-based perspective for network management.** The thesis establishes Graph Neural Networks as a common representational framework for diverse network management problems in optical and wireless domains. Rather than treating restoration, failure prediction, and traffic forecasting as unrelated tasks, the thesis shows that they can all be approached through graph-structured modeling of network entities, dependencies, and operating context.

- **A task-adaptive integration strategy for graph-based intelligence.** The thesis demonstrates that graph-based representations can be combined with complementary methods according to task requirements: reinforcement learning for sequential control, temporal sequence modeling for predictive tasks, and evolutionary optimization for model adaptation. Crucially, this integration extends graph learning beyond static prediction into operational decision support: the restoration and reliability studies in particular consider scalability, runtime, and deployment, strengthening the relevance of the work for real-world network operations.

- **A cross-domain validation of graph learning for resilience, reliability,**

and forecasting. Through three application domains—WDM restoration, 5G radio link failure prediction, and 6G traffic forecasting—the thesis validates the effectiveness of graph-based learning across different layers, technologies, and timescales of network management. These domains correspond to a progression from reactive recovery, to proactive failure anticipation, to predictive traffic-aware planning, demonstrating that graph-centered intelligence applies not to one isolated problem but to a broader class of management tasks and provides a foundation for increasingly autonomous network operation.

1.5 Chapter-Specific Contributions

The chapter-specific contributions of this thesis are summarized as follows:

- **Chapter 3 – WDM Network Restoration:** A DRL+GNN framework is developed for dynamic restoration in optical WDM networks, where network state is encoded as a graph and used to guide topology-aware routing and wavelength assignment after failures. The framework improves recovery effectiveness and demonstrates the value of combining graph-based representation with sequential decision-making for self-healing optical network control.

- **Chapter 4 – 5G Radio Link Failure Prediction:** A graph-based predictive framework is proposed for proactive radio link failure prediction in 5G networks. By combining graph-based spatial modeling with temporal and environmental context, the model improves the early detection of failures and supports proactive reliability management in wireless radio access networks.

- **Chapter 5 – 6G Traffic Forecasting:** A spatio-temporal GNN-based traffic forecasting framework is introduced for 6G-oriented network environments, incor-

porating Transformer-based temporal modeling and GA-based hyperparameter optimization. The resulting model improves forecasting accuracy and supports predictive resource allocation under dynamic traffic conditions.

1.6 Thesis Outline

The remainder of this thesis is organized as follows. Chapter 2 reviews the background and related work that motivate a graph-centered perspective on intelligent network management across restoration, failure prediction, and traffic forecasting tasks. Chapter 3 develops a GNN-enhanced deep reinforcement learning framework for dynamic restoration in WDM networks. Chapter 4 develops a graph-based predictive framework for radio link failure prediction in 5G networks. Chapter 5 presents a spatio-temporal GNN-based framework for traffic forecasting in 6G-oriented environments, augmented with temporal sequence modeling and genetic algorithm-based hyperparameter optimization. Chapter 6 concludes the thesis by synthesizing the main contributions, highlighting the broader implications of graph-based learning for network management, and identifying directions for future work.

1.7 List of Publications

Conference Publications:

- Ampratwum, I., Nayak, A. (2024, July). Optimizing WDM Network Restoration with Deep Reinforcement Learning and Graph Neural Networks Integration. In 2024 IEEE 48th Annual Computers, Software, and Applications Conference (COMPSAC), (pp. 794-800). IEEE

- Ampratwum, I., Nayak, A. (2025, July). Graph Neural Network-Based Internet Traffic Prediction in 6G Networks with Genetic Algorithm Hyperparameter Optimization. In 2025 IEEE 49th Annual Computers, Software, and Applications Conference (COMPSAC) (pp. 743-748). IEEE.

- Ampratwum, I., Nayak, A. (2025, July). Radio link failure prediction in 5G networks using graph neural networks. In 2025 IEEE 49th Annual Computers, Software, and Applications Conference (COMPSAC) (pp. 729-736). IEEE.

Journal Publications:

- Ampratwum, I., Nayak, A. (2025). Hybrid Approach for WDM Network Restoration: Deep Reinforcement Learning and Graph Neural Networks. IEEE Open Journal of the Computer Society, vol. 6, pp. 1012 - 1026.

- Ampratwum, I., Nayak, A. (2026). Predictive Maintenance and Reliability in Intelligent 5G Networks based on Graph Neural Networks. IEEE Access, DOI: 10.1109/ACCESS.2026.3673933.

Chapter 2

Background & Related Works

The increasing complexity and scale of modern communication networks have necessitated a shift from static, rule-based management schemes to more intelligent, adaptive approaches. In optical and wireless domains alike, networks are expected to deliver ultra-reliable, low-latency, and high-throughput services under dynamic and often unpredictable conditions. To meet these demands, recent research has explored the application of machine learning (ML), reinforcement learning (RL), and GNNs to key network management problems such as fault recovery, link failure prediction, and traffic forecasting.

This chapter provides background and a comprehensive review of existing literature relevant to the three primary focus areas of this thesis. First,it provides a conceptual background on WDM networks and on 5G/6G networks, highlighting their architectures, the issues caused by link failures and dynamic traffic, and the motivations for robust restoration and predictive management mechanisms. it then examines the evolution of restoration strategies in optical Wavelength Division Multiplexing (WDM) networks, tracing the transition from traditional protection/restoration heuristics to optimization-based and learning-based methods. Emphasis is placed on the emerging integration of GNNs and deep reinforcement learning for dynamic and topology-aware recovery after

failures.

Next, the chapter surveys the state of the art in radio link failure (RLF) prediction within 5G networks. It outlines early threshold-based and statistical approaches, as well as more recent advances using supervised learning models that leverage radio KPIs and environmental data. Special attention is given to the use of spatial learning methods such as GNNs for aggregating neighborhood context, and the incorporation of temporal sequence modeling for proactive failure anticipation.

Finally, the chapter reviews traffic prediction techniques in 5G and emerging 6G environments. It explores the progression from classical time-series models to deep learning architectures, including LSTMs, CNNs, and Transformers. The section also highlights hybrid models that combine GNNs and attention-based mechanisms to capture spatial-temporal dynamics, along with evolutionary strategies like Genetic Algorithms (GAs) for hyperparameter optimization and model generalization.

By organizing this literature review around the core challenges addressed in this thesis—restoration, failure prediction, and traffic forecasting—this chapter establishes a foundation for the novel contributions introduced in Chapters 3 through 5. The selected works demonstrate both the limitations of conventional approaches and the growing potential of graph-based, learning-driven techniques in enabling intelligent and resilient network management.

2.1 Background

2.1.1 WDM Networks

2.1.1.1 Architecture and Operation of WDM Networks

WDM is an optical networking technology that multiplexes multiple data channels on different wavelengths of light over a single fiber [8].A WDM network typically consists of multiple nodes connected by fiber in a mesh or ring topology [9]. At network nodes, devices like optical add-drop multiplexers (OADMs) and optical cross-connects (OXCs) enable the routing or switching of individual wavelength channels without needing conversion to electronics[9]. Each wavelength/lightpath acts as an independent communication channel, so a single fiber can carry dozens or even hundreds of high-speed channels in parallel (e.g., Dense WDM systems commonly support 40, 80 or more wavelengths). This architecture dramatically increases fiber capacity and allows transparent end-to-end optical circuits, forming the backbone of Internet and telecommunication infrastructures. WDM nodes can be configured to establish end-to-end wavelengths through the network by reserving a wavelength across a sequence of fiber links. Reconfigurable optical add-drop multiplexers (ROADMs) and OXCs provide flexibility to dynamically set up or tear down light-paths, enabling efficient use of wavelengths. Overall, WDM networks operate on principles of optical circuit switching, where connections are typically set up via a control plane and then remain in the optical domain during data transmission. Such all-optical operation yields high throughput and low latency, as signals don't undergo O/E/O (optical-electrical-optical) conversion at intermediate nodes.

2.1.1.2 Link Failures in WDM Networks

While WDM technology brings enormous capacity, it also introduces vulnerability: a single physical link carries many channels, so a fiber cut or node failure can simultaneously disrupt a large aggregate bandwidth. In fact, the failure of a single optical link or node in a WDM network may cause the simultaneous failure of several optical channels [8].Compared to traditional networks, the impact of a single failure is magnified, e.g. if a fiber carried 80 wavelengths each at 100 Gbps, a cut could drop 8 Tb/s of traffic instantly. This phenomenon is sometimes called failure propagation, as the optical-layer failure can trigger multiple higher-layer link failures at once[8]. Because many independent data channels share the same fiber, the amount of bandwidth lost by a resource failure is now much larger than what would have been lost in a traditional network. This has driven the importance of building survivable WDM networks so that a single node or link failure does not interrupt ongoing communications. In practice, WDM backbone networks are engineered for high reliability, but fiber cuts due to construction, disasters etc and device failures do occur. Therefore, robust failure management is critical in WDM networks to maintain service continuity for all the carried channels.

2.1.2 5G and 6G Networks

2.1.2.1 5G Network Architecture and Features

The Fifth Generation (5G) mobile network represents a major architectural evolution from 4G. The 5G system is designed with a cloud-native core network and a flexible, service-based architecture (SBA). In the 5G Core, network functions are decomposed into modular Network Functions (NFs) that communicate via web-like service interfaces[10].This SBA approach enhances modularity and reusability of network services, allowing features like network slicing and dynamic scaling. The 5G New Radio

(NR) access network includes advanced technologies such as millimeter-wave (mmWave) bands, massive MIMO antenna arrays, and beamforming to achieve very high data rates and capacity. In terms of capabilities, 5G was designed to meet IMT-2020 requirements defined by the ITU [11]. Notably, 5G supports three primary usage scenarios:

- **Enhanced Mobile Broadband (eMBB)**: for extremely high data rates and capacity (e.g. gigabit mobile connectivity, UHD video, AR/VR). 5G can deliver peak downlink speeds up to 20 Gbps under ideal conditions while also serving dense crowds with high per-user throughput [11].
- **Massive Machine-Type Communications (mMTC)**: to connect billions of Internet-of-Things devices with low-power, low-cost connections. The network is optimized for very high connection density with low data rates per device [11].
- **Ultra-Reliable Low-Latency Communications (URLLC)**: for mission-critical applications that require near-instantaneous, highly dependable links (e.g. industrial automation, vehicle-to-vehicle safety messaging). 5G targets millisecond-level latency and reliability better than 99.999% for these services [11].

To enable these diverse services, 5G introduced features like network slicing, which allows multiple virtual networks on the same physical infrastructure, each tailored to specific requirements (for example, a low-latency slice for URLLC and a high-throughput slice for eMBB).

2.1.2.2 6G Vision and Enhanced Capabilities

Sixth Generation (6G) networks are still in the research and vision stage (expected around 2030), but early consensus is that 6G will extend and surpass 5G capabilities in several dimensions. The vision for 6G is often described as a fully integrated, intelligent

communications fabric that connects everything ubiquitously. One comprehensive 6G vision suggests that 6G will be based on three fundamental elements: wireless, artificial intelligence (AI), and the Internet of Everything (IoE), converging into an Intelligent Network of Everything [12]. In practice, this means 6G networks are expected to deeply integrate communication with computing, sensing, and control.

Architecturally, 6G will likely continue the softwarization trend of 5G (virtualized NFs, cloud RAN, etc.) and push it further with even more distributed cloud-edge infrastructure and embedded AI for self-optimization. Network slicing will evolve, potentially becoming more dynamic and cross-domain (spanning across terrestrial and non-terrestrial networks). A key aspect of 6G's architecture is the integration of heterogeneous networks: terrestrial cellular networks, satellite constellations, aerial networks (e.g. drones, high-altitude platforms), and even underwater communications are expected to interwork seamlessly [13]. This would provide ubiquitous and unlimited wireless connectivity by covering all environments from dense urban areas to remote rural, to air and space[13]. In terms of performance targets, 6G is aiming for extremes: peak data rates on the order of multi-terabits per second (leveraging sub-THz frequency bands), end-to-end latencies perhaps less than 0.1 ms in some scenarios, and support for extreme mobility up to even faster speeds (e.g., high-speed transport, low earth orbit satellites). Requirements outlined in the IMT-2030 vision include increased coverage, capacity, extremely high user data rates, much lower latency, and high reliability, beyond what 5G can offer [13]. Energy efficiency and sustainability are also highlighted, meaning 6G technology must deliver these gains without a prohibitive increase in power consumption. Another anticipated hallmark of 6G is the pervasive use of AI and machine learning in all layers of the network. The network is expected to have built-in intelligence for tasks like dynamic spectrum access, adaptive coding/modulation, and network management. For

example, 6G may feature an intelligent core and intelligent edge where predictive analytics and real-time optimization are constantly managing resources and QoS. By doing so, 6G hopes to autonomously handle complexity and support new applications such as holographic telepresence, immersive extended reality, and autonomous systems communications. In summary, 6G's capabilities will be an extension of 5G's goals (broadband, massive connectivity, ultra-low latency) to an even more ambitious scale, while introducing integration of sensing, localization, and communication services into one platform [13]. It is envisioned as a network that connects everything, provides full-dimensional wireless coverage, and integrates all functions, including sensing, communication, computing, caching, control, positioning, and even imaging to enable a truly intelligent and immersive digital society [13].

Although 6G is best understood as an extension of 5G, the two generations differ substantially in their quantitative targets. Table 2.1 summarizes the fundamental differences in requirements between 5G, as defined by IMT-2020, and the current 6G vision (IMT-2030) [13].

In essence, 6G does not merely scale up 5G's numerical targets by roughly an order of magnitude; it also broadens the network's scope from a communication system toward an integrated communication, sensing, and computing platform with native intelligence and full-dimensional coverage.

2.1.2.3 Link Failures and Traffic Dynamics in 5G/6G Networks

Despite advanced designs, 5G and future 6G networks are not immune to link failures and performance variability. However, the nature of "link failures" in these wireless networks differs from optical WDM failures. Key failure scenarios and dynamic aspects include:

- **Wireless Link Failures (RAN)**: In 5G radio access, a radio link failure (RLF)

Table 2.1: Fundamental differences in requirements between 5G (IMT-2020) and the 6G vision (IMT-2030).

Requirement	**5G (IMT-2020)**	**6G (IMT-2030 vision)**
Peak data rate	20 Gbps	$\sim$1 Tbps
User-experienced rate	100 Mbps	1–10 Gbps
Latency (user plane)	1 ms	0.1 ms
Reliability	$1 - 10^{-5}$ (99.999%)	$1 - 10^{-7}$ (99.99999%)
Connection density	10^6 /km^2	10^7 /km^2
Mobility support	500 km/h	up to 1000 km/h
Spectrum	mmWave (up to $\sim$52 GHz)	sub-THz / THz
Positioning accuracy	meter-level	centimeter-level
Energy efficiency	baseline	$\sim$10–100$\times$ improvement
Intelligence	AI-assisted	AI-native (built-in)
Coverage	terrestrial-centric	integrated terrestrial & non-terrestrial

refers to the event when a user equipment (UE) loses its connection to the base station (gNB) due to poor signal quality or outage. High-frequency 5G NR links (especially mmWave bands) are more susceptible to blockage (e.g., by buildings or even a human body) and rapid signal degradation. If a user moves behind an obstruction or out of cell range, the radio link can drop unexpectedly. Studies highlight that RLF is a challenging problem in 5G networks as it may decrease communication reliability and increase latency, which is against the objectives of 5G, particularly for the URLLC traffic class [14]. In other words, frequent radio link drops undermine the ultra-reliability and low-latency promises of 5G. Factors like high mobility (e.g., users in vehicles or UAVs) and limited small-cell coverage areas exacerbate this issue [14]. For example, a 5G-connected drone may experience RAN link failures more often due to altitude and sparse coverage [14]. Each RLF triggers reconnection procedures (RRC re-establishment or handover to another cell), during which there is service interruption.

- **Traffic Dynamics and Variability**: 5G and envisioned 6G networks must handle extremely dynamic traffic patterns. The introduction of diverse services (eMBB vs. mMTC vs. URLLC) means traffic on the network can vary widely over time and location. For instance, live video streaming or AR applications can create sudden spikes in bandwidth demand, whereas IoT sensor networks might generate bursts of signaling at specific intervals. Mobility of users (especially in 5G, where handovers between small cells are frequent) causes fluctuating load on different cells over time. Moreover, 5G supports network slicing, where each slice might have its own traffic profile – one slice may experience a heavy uplink data burst (e.g., from IoT devices reporting), while another slice sees steady downlink video traffic. All these factors lead to a situation where network conditions are highly variable and

less predictable than in legacy networks

In summary, 5G/6G networks face two related challenges: ensuring reliability despite more fragile or complex links in the networks and coping with highly dynamic, heterogeneous traffic loads. Link failures in these networks such as a sudden radio link drop can degrade user experience significantly causing dropped calls, interrupted streams, etc. and even compromise safety-critical services. Likewise, unanticipated traffic surges or shifts could overload network components, causing congestion and outages if not handled. These challenges motivate the adoption of advanced, proactive network management strategies in 5G and even more so in 6G.

2.1.3 Graph Neural Networks

Graph Neural Networks (GNNs) are the methodological foundation shared across the three studies in this thesis, and are therefore described here in more detail. A GNN is a class of deep learning model designed to operate directly on graph-structured data, in which entities are represented as nodes and their relationships as edges. Unlike conventional neural networks, which assume inputs are independent, GNNs explicitly exploit the relational structure of the data, making them well suited to communication networks, where routers, base stations, optical nodes, and links form natural graphs [5].

The core mechanism underlying most GNNs is neighborhood aggregation, also known as message passing. Each node maintains a feature vector (embedding) that is iteratively updated by aggregating information from its neighbors. At layer k, the embedding $h_v^{(k)}$ of node v is computed by combining its previous embedding with an aggregation of its neighbors' embeddings:

$$h_v^{(k)} = \text{UPDATE}\big(h_v^{(k-1)},\ \text{AGGREGATE}\big(\{h_u^{(k-1)} : u \in \mathcal{N}(v)\}\big)\big)\,, \tag{2.1}$$

where $\mathcal{N}(v)$ denotes the neighbors of node v, and AGGREGATE and UPDATE are differentiable functions (e.g., a sum, mean, or attention-weighted combination followed by a neural transformation). After K layers, each node embedding captures information from its K-hop neighborhood, letting the model reason about local and increasingly global topological context.

Several architectures instantiate this framework differently. Graph Convolutional Networks (GCNs) aggregate neighbor features using a normalized, degree-weighted sum, generalizing the convolution operation to irregular graph domains [15]. GraphSAGE introduces sampling and general aggregation functions to enable inductive learning on large graphs [16]. Graph Attention Networks (GATs) assign learnable attention weights to different neighbors, allowing the model to emphasize the most relevant connections rather than weighting all neighbors equally [17]. These architectures form the building blocks adapted in the subsequent chapters.

A key strength of GNNs is their combination of permutation invariance, inductive generalization, and scalability. Because aggregation does not depend on any fixed ordering of nodes, GNNs naturally handle graphs of varying size and structure, and a model trained on one set of topologies can in principle generalize to unseen or larger networks, since it learns relational patterns rather than memorizing fixed layouts. This inductive capability is especially valuable in network management, where topologies differ across deployments and evolve over time. GNNs have consequently been applied to routing optimization, performance modeling, fault diagnosis, and resource allocation [5].

Nevertheless, GNNs present limitations relevant to their use in this thesis. Deep GNNs can suffer from over-smoothing, where repeated aggregation makes node embeddings indistinguishable, limiting useful depth. Their performance depends on how well the underlying graph is constructed and on the quality of node and edge features, and

scaling to very large graphs can require sampling or approximation. These considerations motivate the task-specific choices in graph construction, feature engineering, and the integration of complementary temporal and decision-making mechanisms described in the following chapters.

2.2 Unifying Perspective Across Network Management Tasks

Although the three application domains studied in this thesis—optical network restoration, radio link failure prediction in 5G networks, and traffic forecasting in 6G-oriented environments—are often treated as separate research areas, they share a common underlying challenge. In each case, the network must be managed under conditions of structural complexity, temporal variability, and operational uncertainty. The network itself is naturally represented as a graph, with dependencies between nodes, links, and traffic flows that cannot be adequately captured by methods that treat observations as independent or static. This makes these problems fundamentally suitable for structure-aware learning approaches.

From a management perspective, these domains differ in operational objective but not in underlying difficulty. Restoration focuses on recovering from disruption after failure has occurred. Failure prediction focuses on anticipating degradation before service is lost. Traffic forecasting focuses on anticipating future demand so that capacity and resources can be managed proactively. Together, these tasks span reactive, predictive, and anticipatory forms of network management, but all require models that can reason over topology, capture evolving state, and support decision-making under uncertainty.

This chapter reviews the relevant background and literature for each of these areas

through that common lens. Rather than presenting three unrelated surveys, the chapter aims to show that restoration, reliability prediction, and traffic forecasting can all be understood as different manifestations of the broader problem of intelligent network management in graph-structured communication systems. This framing motivates the use of Graph Neural Networks as the unifying representational foundation for the methods developed in later chapters.

The remainder of this chapter reviews the state of the art for each of the three problems addressed in this thesis. These sections provide a common, cross-cutting perspective that situates all three contributions within the broader landscape of learning-based network management and identifies the shared research gaps summarized at the end of this chapter. In addition, each contribution chapter (Chapters 3–5) contains its own focused related-work section that examines, in detail, the specific prior studies most directly relevant to that individual contribution. This division is deliberate: the present chapter establishes what is *common* across restoration, failure prediction, and traffic forecasting, whereas the per-chapter reviews address what is *individual* to each study.

2.3 State-of-the-art WDM Restoration

The first problem considered in this thesis is network restoration, where the challenge is to make effective decisions after a disruption has already occurred. This is the most reactive of the three management settings considered.

2.3.1 Traditional Approaches to WDM Restoration

Early WDM network survivability techniques are broadly categorized into protection (proactive) referred to as $Pro - R$ in this study and restoration (reactive) methods

referred to as $R-R$. $Pro-R$ tackles failure restoration by proactively provisioning $K+1$ paths. This strategy involves pre-computing and reserving a primary path along with K backup paths to ensure service continuity in the event of link failures. The selection of K is typically guided by the network's desired level of resilience and its ability to withstand a specific number of simultaneous faults [18], [19]. Under this scheme, each service is assigned both a primary path and K disjoint secondary paths, where the resources allocated to these paths are proportional to the service demand. In scenarios where link failures occur involving up to K faults, the service can continue operating uninterrupted by switching to an alternate path.

In previous research studies [20], an alternative proactive mechanism known as "p-cycle" has been explored. P-cycles offer survivability by utilizing predefined secondary routes in a cyclic configuration, shared among multiple primary routes. However, it is noteworthy that the effectiveness of this approach is closely tied to the network's scale, as it can potentially introduce considerable additional latency for users in a protection state, particularly in larger networks. Furthermore, implementing multiple fault tolerance using p-cycles necessitates the deployment of a substantial number of such cycles. This proliferation of cycles raises practical challenges and considerations. Reactive restoration ($R-R$) includes 1:1 or shared mesh protection as examined in studies [21], [22], and [23]. These strategies allocate backup resources only when failures occur, improving resource efficiency by sharing paths among multiple services. Backup paths can be precomputed (offline) or dynamically calculated (online). While offline methods reduce computation time during failures, online approaches offer real-time adaptability. These conventional techniques, while effective for single-link failures, often struggle in complex scenarios. They generally assume a predetermined backup for each connection and have difficulty adapting if multiple concurrent failures occur or if traffic patterns change significantly.

Indeed, traditional proactive and reactive schemes optimized for single failures degrade in performance when facing multiple simultaneous outages. Moreover, standard routing protocols (e.g., link-state algorithms) can suffer from slow convergence and high control overhead after a failure, which impedes timely recovery in large-scale optical meshes [24].

In summary, while traditional WDM restoration methods have laid the foundation for network resilience, their limitations include inefficient resource usage (in proactive schemes), higher restoration latency (in reactive schemes), and limited adaptability to unforeseen or multi-failure situations. These shortcomings motivate the exploration of more intelligent and flexible restoration strategies.

2.3.2 Machine Learning and Optimization-Based Techniques

To overcome the rigidity of classical approaches, researchers have explored optimization algorithms and machine learning for network restoration. Optimization-based techniques typically formulate restoration or protection planning as an NP-hard routing and resource allocation problem. Integer Linear Programming (ILP) models can find optimal solutions for small networks, but they become computationally intractable as network size grows [6]. For instance, even networks of only a few tens of nodes lead to ILP formulations that are infeasible to solve in real time [6]. Consequently, ILP-based restoration cannot meet the strict timing requirements of carrier-grade recovery [25].

In practice, operators turn to heuristic and meta-heuristic algorithms that produce near-optimal solutions more quickly. Algorithms such as greedy path selection, genetic algorithms, and other evolutionary approaches have been applied to WDM survivability problems [26]. These heuristics avoid the exponential complexity of ILPs, but they still face challenges in scalability and often require expert tuning. For example, a genetic algorithm-based heuristic was used to schedule repair of failed connections [25], and

threshold-based schemes have been proposed to selectively restore only critical flows under resource constraints. While such methods improve efficiency, they do not guarantee optimal recovery and may struggle to cope with very large or highly dynamic networks.

In recent years, machine learning (ML) techniques, especially reinforcement learning, have been introduced to enhance network restoration and resource allocation. The advent of Software-Defined Networking (SDN) has made it easier to apply centralized ML algorithms for network control. Several works leverage reinforcement learning (RL) to let the network "learn" effective restoration policies through trial and error. For example, in optical networks, a deep Q-learning agent was employed for dynamic routing and spectrum assignment (RSA) in elastic optical networks (Deep-RMSA) [26]. This DRL-based RSA learned to reroute traffic under changing conditions and achieved lower blocking probability than fixed heuristic policies [26]. Similarly, in an SDN context, RL has been used for fast failure recovery – one recent approach (FRRL) can immediately compute a new routing policy when a link in a hybrid SDN network fails [27], enabling near-instant failover. These ML-driven methods offer adaptability: the model can observe the network state (e.g. which link is down, current traffic loads) and output a restoration action (choose an alternate path or reconfigure lightpaths) without pre-programmed rules. Over time, the agent improves its decisions to maximize a reward (such as minimizing disruption or congestion). However, purely RL-based solutions also have limitations. Training such agents for every specific network topology can be time-consuming, and their performance may not generalize if the network changes. Notably, a recent comprehensive study found that in some cases, published RL approaches for optical resource allocation did not significantly outperform well-tuned heuristics [28]. Issues like limited exploration, state space explosion in large topologies, and lack of generalization can hinder standalone ML approaches. These observations highlight the need for more robust learning frameworks

that can scale and generalize better than conventional RL or optimization techniques.

In summary, prior works discussed in this section highlight several traditional and contemporary strategies, including proactive restoration methods with pre-computed routes and reserved resources [18]-[20], reactive restoration that dynamically searches for resources post-failure [21][22], mathematical optimization approaches based on integer linear programming [6] and heuristic algorithms [25], and recently emerging machine learning methods, particularly supervised reinforcement learning [26]. However, each approach presents significant limitations when specifically applied to the unique challenges of the WDM network restoration problem addressed in our study. Proactive methods suffer from inefficient resource utilization due to the pre-allocation of backup resources. Reactive strategies, while more efficient in resource usage, often incur significant restoration delays due to dynamic real-time path computations. Mathematical optimization techniques quickly become computationally intractable for large-scale networks, impeding their feasibility for real-time applications. Reinforcement learning only based approaches give solutions that do not generalize on other networks.

2.4 State-of-the-art Radio Link Failure Prediction

The second problem moves one step earlier in the management timeline: rather than reacting after disruption, the goal is to predict failure risk before service degradation occurs.

2.4.1 Traditional Methods

Radio Link Failure (RLF) in cellular networks (4G/5G) refers to the abrupt loss of the radio connection between a User Equipment (UE) and the base station. Traditional

cellular systems handle RLF in a reactive manner: the network declares a link failure after certain thresholds are breached for example, the UE's Reference Signal Received Power (RSRP) or Quality (RSRQ) falls below a set level for a sustained duration as defined by 3GPP standards. These threshold-based triggers and associated timers ensure that RLF is detected, but they only indicate failure after the link has already degraded badly. Proactively predicting RLF before it happens is challenging because it depends on multiple time-varying factors such as signal quality, interference, UE mobility, and it is difficult to derive an exact analytical model mapping those factors to a future failure event [14]. In fact, prior work noted that while one can measure the instantaneous radio link quality and even estimate the current probability of outage, finding a closed-form model for the chance of an impending RLF is extremely difficult [14]. This realization has driven interest in data-driven approaches that learn the complex patterns leading to RLF.

2.4.2 Machine-learning-based Methods

Initial research on RLF prediction often tied into improving handover decisions, since many link failures occur when a moving UE should have handed over to a better cell but did not in time. For example, one study applied deep learning to predict handover outcomes: using an LSTM (Long Short-Term Memory) network on the time series of serving-cell and neighbor-cell RSRP measurements, they predicted whether a planned handover in the next 1–2 seconds would succeed or result in a failure. This helped reduce RLF by prompting earlier handovers for UEs likely to drop. Another work took a more network-centric approach by clustering cells based on similarity in handover performance; cells with frequent handover failures were grouped and an early handover trigger policy was applied to those clusters to preempt RLF. Both approaches showed benefits in simu-

lation, though they were focused on specific scenarios which is intra-network mobility and might not generalize to all causes of RLF. Beyond handover-related failures, researchers have tackled general RLF prediction using machine learning classification and regression techniques. These methods typically ingest recent radio measurement reports (RSRP, RSRQ, Signal-to-Interference ratios, etc.), possibly along with context like UE speed or environment, and output a prediction of whether the link will fail in the near future.

Tree-based models have been popular for their interpretability and speed: Agarwal et al. [29] used decision trees and Random Forest classifiers to predict link failures up to five days in advance. They engineered features from both the radio KPIs and external weather data, under the observation that environmental factors (rain, humidity) can degrade signal quality in mmWave and higher frequencies. In their implementation, each UE-link was associated with data from its closest weather station, and a Random Forest model was trained to classify if a link would fail within a certain horizon. The inclusion of weather significantly improved reliability, confirming that harsh weather can be a strong predictor of RLF in 5G.

Aktas et al. [30] took a different approach by designing a branched neural network: one branch was an LSTM processing the time-series of recent KPI measurements, while the other branch was a feed-forward network ingesting static or categorical features such as cell ID or terrain type. The outputs of these branches were combined to predict RLF, achieving higher accuracy than single-branch models.

An interesting anomaly-detection style approach by Islam et al. [31] trained an LSTM-based autoencoder on "normal" connectivity data (periods with no failure) and then declared an impending failure whenever the model's reconstruction error for the latest sequence of KPIs exceeded a threshold. Essentially, their model learned to recognize the normal patterns of signal variation; a large deviation reconstructed by the autoen-

coder signals a likely abnormal event such as an upcoming drop. Such unsupervised methods are valuable when explicit failure labels are scarce.

Deep learning, particularly neural networks, have demonstrated very high accuracy in RLF prediction when sufficient data is available. For instance, Boutiba et al. [14] combined an LSTM with an SVM classifier in a hybrid model and trained it on real 5G testbed data. The LSTM was used to forecast the near-future KPI values (RSRP, RSRQ, etc.), and then the SVM classified whether those forecasts indicate a pending RLF. This model achieved about 98% accuracy in predicting connectivity status (failure or not) on the testbed. Similarly, other studies report high recall (e.g., ¿95%) in identifying upcoming failures, meaning most failures can be successfully predicted a short time before they occur. These results are promising, as they suggest ML models can learn subtle pre-failure signatures in the data (such as a characteristic decline in RSRQ coupled with a power headroom drop, etc.).

In summary, RLF prediction in 5G has evolved from threshold-based reactive schemes to proactive ML-driven approaches. Threshold and rule-based methods are simple but have inherent limitations: they can miss complex failure precursors and generally cannot provide advance warning beyond very basic scenarios. Conventional ML models like decision trees and SVM brought in the ability to handle non-linear combinations of metrics and gave some early success in prediction, though they might require significant feature engineering e.g., combining signals with domain knowledge. Deep learning models improved prediction lead time and accuracy, discovering patterns that humans might not anticipate. However, each model had assumptions: LSTMs assume temporal patterns are the key with less focus on exogenous data, while some early works assumed failures mostly occur around handovers or specific events.

2.5 State-of-the-art Traffic Prediction in 5G/6G networks

The third problem extends this shift toward proactive management by forecasting future traffic demand, enabling resource planning before congestion or under-provisioning becomes critical. Accurate traffic prediction in telecommunication networks has long been pursued to enable proactive resource management, from dynamic capacity provisioning to intelligent routing and load balancing. In the context of 5G and emerging 6G networks, traffic patterns are becoming more complex due to heterogeneous services (eMBB, URLLC, mMTC) and the interplay of wired and wireless segments. The literature on network traffic forecasting spans traditional statistical methods to modern deep learning, evolving as the complexity and volume of data increased.

2.5.1 Traditional Statistical Methods

Early works applied time-series models such as ARIMA (AutoRegressive Integrated Moving Average) to network traffic data. ARIMA and its seasonal variants (SARIMA) model the traffic as a combination of trends, seasonality, and noise, and have been used successfully for short-term traffic load predictions [32]. For example, researchers have used ARIMA to forecast LTE base station traffic over minutes to hours, finding it reasonably accurate for regular patterns [33]. Other statistical approaches include exponential smoothing (Holt-Winters) for capturing seasonal effects, and linear regression models for trend extrapolation [32]. These methods are appealing due to their simplicity and low computational cost which is important for real-time forecasting in early networks. In fact, one study demonstrated that a well-tuned SARIMA model could achieve a Mean Absolute Percentage Error (MAPE) around 10% on cell traffic, comparable to some

more complex methods [32]. However, statistical models make strong assumptions e.g., that the underlying process is stationary or that past patterns will repeat. They often struggle with non-stationary behavior common in modern networks for instance, abrupt changes due to a viral app or an exceptional event. Moreover, while fast, they typically provide one-step-ahead forecasts and may degrade in accuracy for longer horizons unless extended with exogenous inputs.

2.5.2 Machine-learning-based Methods

To handle non-linear patterns in traffic data, the community introduced machine learning algorithms. Techniques such as Random Forests, Support Vector Machines (SVM), Gaussian Process Regression, and even simple feed-forward neural networks were applied to predict network traffic load [32]. These models can capture complex relationships that ARIMA might miss like day-of-week effects or holiday anomalies by learning directly from data. Empirical studies showed that ML models often outperform purely statistical ones in terms of prediction accuracy on real network traces [32]. For instance, a Random Forest might automatically leverage dozens of input features like previous lags, time-of-day and interactions between them, improving accuracy. That said, classical ML models have their own limitations. Many require feature engineering (e.g., creating input features like "traffic from same hour yesterday" or "average of last 3 hours"), and they don't inherently capture temporal dynamics beyond what is fed in. To address this, some researchers created hybrid models: one approach used an ML to first predict the traffic and then an ARIMA model to predict the residual error of the ML model, effectively combining strengths [34]. The key point is that as traffic patterns grew more complex, pure statistical models became less adequate, paving the way for data-driven learning methods that could ingest large volumes of historical data.

2.5.3 Deep Learning Approaches

In the last decade, deep learning has revolutionized traffic prediction, mirroring its success in other time-series domains. Recurrent Neural Networks, especially LSTMs and GRUs, have been widely adopted for their ability to capture long-term dependencies and sequence patterns. Researchers have reported that LSTM-based models achieve lower error than ARIMA on various network traffic datasets, particularly because they can naturally model the periodicity and burstiness in traffic. For example, a study forecasting 5G base station traffic using GRU and LSTM networks found these models could automatically learn daily patterns and unexpected surges, outperforming traditional models. The authors noted LSTM was effective even for online (real-time) forecasting scenarios [35].

Another advantage of deep learning methods is their flexibility to incorporate multiple features: an LSTM can be fed not only past traffic volumes but also other correlated time series such as number of active users, or external data like day type, which it can learn to utilize. In addition to RNNs, Convolutional Neural Networks (CNNs) have also been applied to traffic prediction. Though traffic data is 1-dimensional time series, CNNs can detect local patterns or motifs in the time series like a recurring spike-and-drop pattern indicating a peak hour followed by a quiet period. By using convolution filters over time lags, a CNN can act as an automatic feature extractor, identifying shape patterns that correlate with future traffic changes. Some works use 1-D CNNs on the traffic sequence to capture short-term correlations and then either directly output a prediction or feed those features into a recurrent network (CNN-LSTM hybrid). Hybrid models have indeed become popular. For example, a ConvLSTM model can refer to a convolutional LSTM network that combines convolutional layers (capturing spatial or short-term features) with LSTM units (capturing longterm dependencies). These hybrids have shown superior

performance. One recent study [36] proposed a model called ConvLSTMTransNet that integrates CNN, LSTM, and even a Transformer encoder in one architecture. By doing so, it captures spatial-temporal relationships in the traffic data and reportedly outperformed baseline models like plain RNN, LSTM, or GRU by about 10% in prediction accuracy. The CNN component in such a model could extract localized trends, the LSTM captures sequential memory, and the Transformer layer adds the ability to focus on important time points (via attention). The result is an architecture that can handle the complexity of real internet traffic, which might have weekly cycles, sudden events, and spatial correlations between different network nodes.

2.5.4 Transformers and Attention Mechanisms

Inspired by their success in other domains, transformer-based models have been introduced to network traffic forecasting as well. Transformers excel at capturing long range dependencies without the vanishing gradient issues of very deep RNNs. They can look at an extended history of traffic: many days or months and potentially learn what distant past patterns are relevant to the current prediction. Researchers have found that attention-based models can outperform RNNs for long-term forecasts, as they can learn to ignore irrelevant past data and emphasize the important parts [37]. For instance, a fully attention-based transformer model was applied to mobile network base station traffic and achieved state-of-the-art results for multi-step forecasting [37]. These models are especially useful when traffic exhibits long seasonal periods or when we want to forecast far into the future (e.g., several days ahead), where RNNs typically accumulate error. The limitation, however, is that transformers have high computational complexity, quadratic in sequence length for vanilla transformers, though research is ongoing on efficient variants like Informer and LogTrans. For 6G scenarios with massive data,

this could be a concern, but approaches like sparse attention and decomposition using transformer for coarse trends and simpler models for fine detail are being studied.

The evolution of network traffic forecasting reflects the increasing complexity and dynamism of modern communication systems, particularly in the transition from 5G to 6G. Early statistical models such as ARIMA and Holt-Winters provided efficient and interpretable short-term predictions, but struggled with non-stationary patterns and long-term accuracy. The introduction of classical machine learning techniques, including Random Forests and Support Vector Machines, enabled the modeling of more complex, non-linear traffic behaviors, though they often required extensive feature engineering and lacked temporal modeling capabilities.

The advent of deep learning marked a significant leap forward, with LSTM, GRU, and CNN architectures achieving superior predictive accuracy by learning long-term dependencies and recurring traffic motifs. Hybrid models combining CNNs with RNNs, and more recently, architectures incorporating Transformer encoders, have further enhanced performance by capturing both short-term variations and long-term trends. Transformer-based approaches, with their attention mechanisms, offer promising results for multi-step and long-range forecasts, though their scalability remains a challenge in large-scale 6G scenarios

2.6 Summary and Unifying Research Gaps

The reviewed literature shows that optical restoration, radio link failure prediction, and traffic forecasting are usually studied in separate technical communities, with different performance metrics and operational assumptions. However, when examined comparatively, they exhibit a shared set of foundational challenges that motivate a common learning-based perspective.

First, all three problems are inherently topological. In optical restoration, decisions depend on the connectivity of the network, the state of links, and feasible rerouting paths under wavelength constraints. In radio link failure prediction, the reliability of a given link is influenced not only by its local KPIs and environmental conditions but also by its spatial and structural relationship to neighboring sites. In traffic forecasting, future demand is shaped not only by temporal history but also by the interaction structure of the underlying network and the relationships between traffic flows. In each case, network behavior is not independent across entities, but instead depends on graph-structured relationships.

Second, all three domains are dynamic. Restoration must respond to changing failure states and resource availability in real time. Failure prediction must account for time-varying weather, KPI behavior, and evolving operating conditions. Traffic forecasting must capture temporal fluctuations, demand trends, and evolving network usage patterns. These are therefore not static optimization problems, but problems of learning and decision-making under changing network state.

Third, all three domains operate under substantial uncertainty. In restoration, the occurrence, location, and effect of failures may not be known far in advance. In radio link prediction, the interaction between environmental factors and network performance is complex and not perfectly observable. In traffic forecasting, future demand is inherently uncertain and influenced by multiple exogenous and endogenous factors. This uncertainty makes purely rule-based or fixed analytical approaches difficult to scale and generalize.

Fourth, the literature indicates a common need for structure-aware learning. Traditional machine learning models and heuristic methods often operate on flattened feature vectors or hand-crafted rules, which can miss the structural dependencies central to network behavior. While sequential models such as LSTMs can capture temporal effects,

they generally do not explicitly encode relational topology. Conversely, graph-based methods provide a natural mechanism for incorporating network structure into learning, making them especially attractive across these domains.

Finally, the literature reveals a shared need for proactive and adaptive decision support. Restoration requires rapid adaptation after disruption. Failure prediction requires early warning before disruption. Traffic forecasting requires anticipation of future conditions before congestion or performance degradation emerges. These are different operational expressions of the same broader requirement: network management must move beyond static configuration and toward adaptive, predictive intelligence.

Taken together, these observations motivate the central perspective of this thesis: although the three application domains differ in objective and timescale, they can all be viewed as manifestations of intelligent management in graph-structured, dynamic, and uncertain systems. This commonality justifies the graph-centered methodological direction adopted in the remainder of the thesis. The following chapters therefore build on this comparative foundation by developing task-specific graph-based frameworks for restoration, reliability prediction, and traffic forecasting.

Chapter 3

Hybrid Approach for WDM Network Restoration: Deep Reinforcement Learning and Graph Neural Networks

This chapter addresses the first of the three network management settings considered in this thesis: *reactive management*. Specifically, it investigates how graph-based learning can support rapid recovery after network disruption in optical WDM systems. Within the broader dissertation argument, this chapter demonstrates that when network management requires real-time control under failure, graph representation provides a strong structural foundation, while deep reinforcement learning becomes necessary to support sequential decision-making over topology-aware restoration actions.

3.1 Introduction

In an era of exponential digital growth, ensuring the resilience of high-capacity networks such as WDM networks has become a critical challenge. These networks, which enable the simultaneous transmission of multiple optical signals over a single fiber, form the backbone of modern internet infrastructure. As the foundation for global communication, WDM networks play a pivotal role in supporting bandwidth-intensive applications, including video streaming, cloud computing, telemedicine, and smart city infrastructures. In WDM networks, the transmission spectrum is divided into distinct, non-overlapping wavelength bands, with each band serving as an independent communication channel capable of operating at a user-defined rate [38]. This architectural design has enabled WDM networks to scale efficiently with growing demands. However, this scalability has also introduced new challenges in ensuring reliability, particularly in the face of unforeseen disruptions.

As the demand for bandwidth continues to soar, the resilience of WDM networks against disruptions, particularly link failures, has emerged as a pressing concern. Link failures can occur due to various factors, including physical damages, natural disasters, or power outages, and their impact can be devastating. Paper [39] emphasizes this issue, citing an average of one fiber cut every five days within networks like NSFNET [40], leading to significant cumulative downtime. Such incidents lead to significant cumulative downtime, with ripple effects across essential services. Failures in these networks can cause widespread service outages, severely impacting critical sectors like telecommunications, financial systems, healthcare, and cloud computing. These disruptions underscore the importance of designing robust network restoration strategies capable of minimizing downtime and maintaining service continuity under adverse conditions.

Traditional restoration strategies within WDM networks are broadly categorized into

proactive and reactive approaches. Proactive strategies involve the pre-computation of restoration paths and the reservation of wavelengths to ensure guaranteed recovery. While this approach enhances network reliability, it does so at the expense of high resource consumption, often leading to increased blocking probabilities. In contrast, reactive strategies allocate spectrum only after a failure has occurred, making them more resource-efficient. However, this delayed allocation can result in prolonged restoration times as the system searches for available resources. Some reactive strategies attempt to mitigate these delays by precomputing multiple restoration paths without reserving spectrum, allowing for faster recovery in certain scenarios. Despite their effectiveness, these traditional methods often fall short in adapting to the dynamic and complex nature of modern networks, necessitating the exploration of more intelligent and adaptive solutions.

The restoration problem in WDM networks has unique characteristics that make traditional methods inadequate and necessitate innovative solutions:

- **High-dimensional Graph-structured Data**: WDM networks inherently represent high-dimensional data structured as graphs, characterized by complex topologies and interdependent link/node relationships. This structure challenges traditional machine learning methods that cannot effectively exploit such relational and topological data.

- **Real-time Decision-making Under Uncertainty**: Restoration decisions must be made rapidly (milliseconds) to minimize network downtime, yet these decisions must accommodate the dynamic and unpredictable nature of network failures. Traditional methods either rely on static, precomputed backup routes (proactive methods) or resource-intensive real-time path searches (reactive methods), neither of which optimally balance speed and adaptability.

- **Complex Combinatorial Optimization**: The problem involves simultaneously selecting optimal restoration paths and wavelengths, introducing combinational complexity due to the vast search space (paths × wavelengths). Efficiently exploring and optimizing this combined search space under tight temporal constraints is a significant challenge, particularly for larger network topologies.

- **Generalization to Unseen Network States**: The solution must generalize effectively across diverse network topologies and states not encountered during training. Classical algorithms typically lack this generalization capability and need extensive recalculations when network configurations change or failures occur unpredictably.

Recent advancements in artificial intelligence, particularly in DRL and GNN, offer transformative potential for addressing these challenges. DRL has proven highly effective in dynamic decision-making tasks by leveraging past experiences and real-time data, while GNNs excel in processing graph-structured data, enabling generalization across diverse network topologies [41]. Together, these AI techniques can revolutionize failure recovery in WDM networks. This chapter proposes a hybrid DRL+GNN framework designed to optimize restoration path selection and wavelength allocation. The DRL component enables real-time decision-making for resource allocation, while the GNN enhances the framework's ability to generalize to unseen topologies, ensuring robust performance in various scenarios. This integration achieves faster, more efficient recovery and addresses the limitations of traditional restoration strategies.

The primary contributions of this chapter, clearly highlighting the novel advancements over existing related studies, are summarized as follows:

- **Novel Hybrid DRL+GNN Framework for Real-Time Restoration**: We propose a novel hybrid framework that uniquely integrates DRL with GNNs specifically tailored for real-time wavelength assignment and path selection in WDM

network restoration scenarios. Unlike traditional proactive/reactive methods and recent learning-based models, our framework adapts dynamically and rapidly to network failures while balancing resource efficiency and speed.

- **Generalization and Scalability**: Our proposed DRL+GNN model demonstrates exceptional generalization capability, effectively applying knowledge gained from training scenarios to previously unseen network topologies.

- **Detailed Computational Complexity and Runtime Analysis**: Unlike prior studies that typically omit detailed runtime and complexity analysis, we explicitly present comprehensive computational complexity and runtime evaluations. Our results confirm the feasibility and practicality of our method, showcasing fast inference times suitable for real-time restoration in operational networks.

- **Rigorous Multi-Topology Experimental Validation**: We significantly extend validation beyond previous DRL or GNN-based studies, rigorously testing our model across multiple real-world and widely recognized network topologies (NSFNET, GEANT, EON, USA). Our extensive evaluation clearly demonstrates consistent performance improvements in restoration probability and resource usage efficiency over established proactive and reactive restoration schemes.

- **Combined Proactive and Reactive Restoration Capability**: Our hybrid approach addresses both proactive (pre-computed restoration) and reactive (dynamic, on-demand rerouting) restoration scenarios within a single unified framework. This dual capability differentiates our solution significantly from existing methods, which typically focus exclusively on one strategy or the other.

- **Adaptability to Dynamic Network Conditions**: Our model inherently captures real-time dynamic changes in network conditions through GNN-based graph

embeddings and adaptive DRL-based decision-making, significantly outperforming traditional heuristic and optimization methods that lack such adaptability.

3.2 Related Works

This section reviews the prior work most directly relevant to WDM network restoration. It complements the broader, cross-cutting survey of learning-based network management in Chapter 2, focusing specifically on the restoration approaches that motivate the DRL+GNN framework developed in this chapter.

3.2.1 DRL and GNN-Based Solutions

DRL combined with GNNs has emerged as a promising hybrid approach to address the above challenges. The key insight is that a communication network can be naturally modeled as a graph (nodes as routers/switches and edges as fiber links), and GNNs excel at learning representations of graph-structured data. By integrating GNN-based state representation into a DRL agent, the restoration strategy can leverage topological awareness and inductive generalization. Unlike traditional neural networks (e.g. fully-connected or CNNs) that require fixed-size inputs, GNNs can process arbitrary network topologies and variable traffic states. This enables a DRL agent to be trained on one set of network scenarios and then generalize to new network sizes or layouts without retraining [41]. In other words, the GNN provides an embedding of the current network state (including link failures and capacities) that is invariant to the network's specific topology size, allowing the DRL agent's decision-making to scale to different or larger graphs. This approach directly tackles the scalability and adaptability issues: the learned policy is not tied to a single network and can adapt to dynamic changes (like additional

node failures or traffic shifts) by extracting up-to-date graph features.

Several recent works demonstrate the efficacy of DRL+GNN in networking problems. Li et al. introduce DRL-GS, a DRL-based graph search method for network topology optimization that tightly integrates a GNN for evaluating candidate topologies [42]. Their framework compresses the action space (possible topology modifications) and uses a GNN to rapidly estimate the network performance of each action, which markedly improves decision speed and scalability [42]. Experimental results show that such a DRL+GNN approach outperforms heuristic algorithms in optimizing network metrics (like link utilization and latency) and scales effectively from small to large network scenarios [42].

In [43], the researchers integrated GNN with Deep Reinforcement Learning (DRL) to emphasize GNN's ability to generalize across diverse network configurations. Their DRL+GNN-based framework demonstrated robust performance in allocating traffic demands within Optical Transport Networks (OTNs), even when applied to topologies that were not included during the training phase. This capability highlights the effectiveness of combining GNN's structural learning with DRL's dynamic decision-making in handling real-world network scenarios.

The authors in [44] advanced the application of DRL by developing two methods aimed at evaluating how design choices impact DRL's performance in solving RWA challenges. Their research explored the generalization of DRL to unseen traffic matrices, examining variables such as the total number of connection requests and traffic distribution patterns. Furthermore, they assessed the scalability of DRL in relation to topology size and link capacity, providing valuable insights into its adaptability for larger and more complex networks.

In [45], the researchers presented a generalized reinforcement learning (RL) framework designed for multi-objective optimization in optical networks. Their model addressed pa-

rameter optimization tasks while considering multiple performance goals simultaneously. Using this framework, they derived two multi-objective variants of the classical RWA problem and achieved near-optimal solutions. This study demonstrated the potential of RL-based approaches to balance multiple objectives, such as throughput, latency, and resource utilization, making it a valuable tool for real-world optical network operations.

In the context of routing, other researchers have proposed GNN-empowered DRL schemes to enhance load balancing and QoS in SDN environments. For instance, a recent intelligent routing method replaces the DRL agent's conventional neural network with a GNN, enabling the agent to better learn over varying topologies and traffic conditions [46]. This GNN-DRL routing model generalizes its experience to different network graphs and was shown to reduce congestion and delay compared to non-learning or non-GNN baselines [46]. These studies underscore that combining GNNs with DRL yields policies that are not only efficient but also transferable across network environments, addressing a weakness of earlier ML-based solutions.

In [47], a Routing and Wavelength Assignment (RWA) scheme for lightpath restoration in WDM networks is proposed, utilizing an active multi-backup path approach. The author suggests deriving successive backup lightpaths along the primary path up to the last node until an available backup lightpath is identified. This method operates without wavelength converters in the network. The algorithm is based on an iterative implementation of Dijkstra's algorithm to find a backup path for the failed primary lightpath, with the source-destination pair (s,d) and the network cost matrix serving as inputs to the routing process.

In [48], the author introduces the GDQR and GD3QR routing algorithms, which combine DRL and GNN to enhance network QoS in complex traffic environments. The research focuses on addressing routing challenges like traffic fluctuations and value over-

estimation through advanced reinforcement learning techniques, including Dueling Network and Double DQN. The GDQR algorithm optimizes routing by modeling both node and link features in the network, while GD3QR further improves on this by mitigating traffic fluctuation impacts and reducing overestimation errors.

The paper [49] presents a novel framework combining DRL and GNN to address the challenges of failure restoration in WDM networks. The study emphasizes the critical importance of WDM network survivability due to their vulnerability to disruptions like link failures, which can lead to significant service outages. Traditional restoration strategies, including proactive and reactive approaches, are resource-intensive or time-inefficient, necessitating the development of more adaptable and intelligent solutions. The proposed DRL+GNN approach leverages the decision-making capabilities of DRL and the structural learning properties of GNNs to optimize restoration path selection and wavelength allocation dynamically. The framework integrates GNN to model the network's dynamic state and DRL to select viable paths, even for previously unseen network topologies.

Reinforcement Learning (RL), particularly DRL combined with GNN, is uniquely suited to addressing these limitations. DRL directly learns restoration policies through ongoing interactions with the environment, adapting dynamically and efficiently to real-time network states without pre-labeled datasets. Combined with GNN's capability to effectively process and learn representations from graph-structured data inherent in WDM network topologies, the hybrid DRL+GNN approach effectively generalizes to diverse and unseen network scenarios, adapts rapidly to dynamic conditions, and significantly reduces restoration times while maintaining efficient resource utilization. This comparative analysis underscores the rationale behind adopting our proposed DRL+GNN framework.

3.3 Graph Neural Network and Deep Reinforcement Learning

3.3.1 Graph Neural Networks

GNNs [40] are a novel class of neural networks designed to operate on data structured as graphs. These networks offer a unique perspective by iteratively associating latent features with neighboring elements, which can be nodes or edges within the graph. This iterative process is commonly referred to as message passing, as outlined in the aforementioned work. Through successive iterations of message passing, GNNs generate a comprehensive representation of the input graph, which can be harnessed to tackle intricate combinatorial optimization problems [50]. The message passing step involves:

- Each node j receives messages from all the nodes in its neighbourhood, denoted by $N(j)$. The neighborhood of a node typically consists of the nodes directly connected to it in the graph.

- Messages are generated by applying a message function m to the hidden state of node pairs in the graph to determine how the hidden states of the connected nodes contribute to the messages being passed.

- Node j then combines all the messages received using an aggregation function such as a sum using the equation (3.1) into a single message.

- Finally, an update function takes into account the current hidden state of the node and the aggregated messages from the neighbors to calculate a new hidden state using equation (3.2).

$$M_j^{t+1} = \sum_{i \in N(j)} m(h_j^t, h_i^t) \tag{3.1}$$

$$h_j^{t+1} = u(h_j^t, M_j^{t+1}) \tag{3.2}$$

Various GNN variants have been developed to suit the specific characteristics of graph nodes and edges, leading to impressive performance in scenarios where data naturally conforms to a graph structure [50]. Given that WDM networks are often represented as graphs, it is reasonable to leverage GNNs as an approach likely to yield favorable results.

3.3.2 Deep Reinforcement Learning

In the domain of deep reinforcement learning, an agent engages with an environment usually represented by Markov Decision Process (MDP), in order to learn a policy that maximizes the expected sum of rewards over a time horizon [51]. The objective is acquiring a strategy that maximizes a predefined objective function. This interaction involves the agent executing actions and observing the resultant feedback from the environment, as elucidated by [51]. The environment's feedback encompasses both the current state of the environment and a reward signal, which serves as an indicator of the quality of the agent's actions. The primary aim of the agent is to determine a strategy that maximizes the cumulative reward over the course of each episode. Given an initial state $s \in S$, the agent selects an action $a \in A$, triggering a transition to a new state $s' \in S$ while receiving an associated reward r.

Q-learning, introduced by [52], is a well-established reinforcement learning algorithm focused on estimating Q-values, representing the quality of actions. This algorithm constructs a Q-table that stores Q-values for all possible state-action pairs. At the start of the training, the Q-table is initialized. It can be filled with zeros or random values. During training, the agent updates these values based on state-action rewards. These Q-values are the predicted cumulative reward after applying action a from state s, assuming that the agent follows the current policy pi during the training episode. During training,

Q-learning uses the Bellman equation (3.3) to update the Q-values where $Q(s_t, a_t)$ is the Q-value function at time-step t, α is the learning rate, $r(s_t, a_t)$ is the reward obtained from selecting action a_t from state s_t and $\gamma \in [0,1]$ is the discount factor. At each state, relying on these Q-values, the agent selects the action with the highest Q-value. However, practical optimization problems often entail large Q-tables.

$$\begin{aligned} Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \\ \alpha[r_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t)] \end{aligned} \tag{3.3}$$

To address this challenge, Deep Q-learning (DQN), as introduced by [53], leverages deep neural networks to approximate Q-values, eliminating the need for a cumbersome Q-table. The high generalization capabilities inherent in deep neural networks enable the agent to approximate Q-values for states not encountered during training. Consequently, our approach is grounded in DQN, capitalizing on its ability to handle complex environments and generalize across unobserved states.

3.4 Proposed Methodology

This study proposes a novel approach combining DRL and GNN for optimal wavelength assignment during link failures. The method addresses both pre-computed restoration routes and real-time routing and assignment scenarios. GNN models the dynamic network state, enabling generalization over graph-structured data [54], while the DRL agent uses this model to make optimization-driven decisions. Figure 3.1 illustrates the overall architecture of the proposed model. The network topology is first represented as graph-structured data, which is then passed to the combined DQN–GNN agent. Within the agent, the GNN encodes the graph-structured network state, and the DQN leverages this

encoding to select the restoration actions.

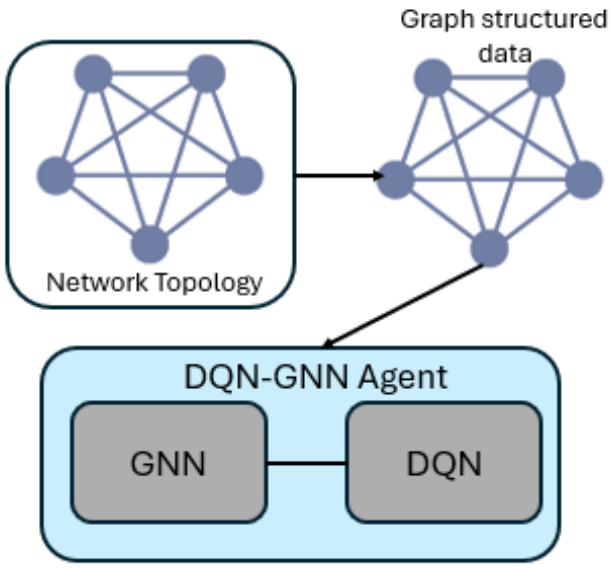


Figure 3.1: Proposed Model

3.4.1 Reinforcement Learning Environment

In this section, we elaborate on the learning environment designed to simulate the failure restoration problem. Our scenario revolves around an SDN-based network, where the DRL agent possesses a comprehensive network overview. When a network failure occurs, the DRL agent's primary task is to select the most suitable pre-computed route for restoration. Furthermore, the DRL agent not only determines the routing but also assigns wavelengths to the affected traffic demands. To create a realistic modeling environment, we maintain a steady state in the network, where a random assortment of demands is routed and assigned wavelengths using both the shortest path forwarding and first-fit algorithms. We also simulate various single link failure instances. During such a failure event, the agent receives the network state, represented as a graph. The GNN processes this graph, transforming it into a new representation in which the links assume the role

of graph components. This revised representation, containing the route actions under consideration, is subsequently fed into the neural network to approximate the Q-values ($Q(s, a)$) for all feasible actions.

A request, representing a demand affected by a failure event, encompasses both source and destination information. The characterization of the network state hinges on topology link features, which consists link betweenness [55], wavelength slot occupancy, link capacity and wavelength usage. Link betweenness, a concept rooted in graph theory, provides insights into the potential utilization of a link by quantifying how many paths could potentially traverse it [55]. To compute the link betweenness for each link, we divide the total number of end-to-end paths traversing that link by the overall number of paths considered. In our research, we specifically compute four end-to-end paths for each A-Z pair within the network. Link capacity quantifies the overall count of available wavelengths on the respective link. It is worth noting that both link betweenness and link capacity are scalar values, whereas wavelength slot occupancy is represented as a binary vector. This binary vector is sized to match the total number of wavelengths on a given link, where '1' signifies an occupied slot, and '0' indicates a vacant one. Wavelength usage is also a vector feature that gives the availability of a particular wavelength in the network. We compute the wavelength usage by dividing the total number of links that have this wavelength occupied divided by the total number of links in the network. This feature provides the agent information on how likely a wavelength will be available on multiple links.

To define the action space, we adopt a deliberate approach of narrowing it down to a smaller subset, effectively reducing the number of Q-values that require computation. A comprehensive consideration of all potential paths connecting every pair of nodes within the network would lead to an expansive action space, even for relatively modest net-

Table 3.1: Input features of each link

Feature	Description	Type
$f1$	link wavelength capacity	binary vector
$f2$	betweenness centrality	scalar
$f3$	number of available wavelengths	scalar
$f4$	wavelength usage	vector
$f5$	action (route and wavelength)	vector

works. The vast number of link combinations available to create a path between a source and destination further compounds the complexity. Furthermore, each resulting path introduces N potential actions based on different path-wavelength combinations, where N represents the available wavelengths. To manage this complexity, we strategically constrain the action space to a predefined subset of k paths, obliging the agent to select from this limited set of options. Upon the arrival of a request, the agent is tasked with selecting a path from the provided subset and subsequently assigning a wavelength. It is important to note that the agent also has the option to reject the request for restoration. In this context, the number of potential actions, given k paths and N wavelengths, becomes $Nk + 1$. In our specific study, we consider a set of 4 paths and 10 wavelengths, resulting in a total of 41 possible actions within our defined action space. The values for number of paths and wavelengths are chosen to provide a balance between restoration flexibility and computational complexity. They provide options for restorations while managing the complexity for experimental purpose. As previously stated, the steady state network has the demands routed using the shortest path. We therefore select a set of next 4 shortest paths by hops for each source-destination pair.

We introduce the action as an input feature into the network state as done in [56]. The incorporation of action is a pivotal aspect of the agent's design, and it is introduced into

the network state in the form of a graph. This representation ensures that the action remains invariant to node and edge permutations, a crucial factor that enhances the generalization capabilities of the trained GNN. The action is integrated into the network as a link feature. When a link is part of the selected route, a corresponding vector is assigned. This vector is structured in a way that designates the chosen wavelength slot with '1', while marking free slots with '0'. Table 3.1 gives the list of input features used in our work.

During a failure event, our primary objective is to maximize the count of restored demands. To effectively capture this behavior in our modeling, we assign a positive reward of '1' each time the agent successfully fulfills a restoration request. In cases where the agent rejects a request, a_{reject}, the reward is set to '0'.

3.4.2 Graph Neural Network Architecture

The neural network architecture we have adopted comprises two key components: the Graph Neural Network (GNN) and a fully connected Deep Neural Network (DNN). The GNN model is centered around a message passing neural network framework. Leveraging the available link features, the GNN undergoes M message passing iterations. During this iterative process, each node is systematically processed, with a focus on aggregating information from its neighboring nodes while also assigning weights to each neighbor based on their respective significance. The outcome of this operation results in the generation of *messages*. These computed messages are subsequently aggregated using an element-wise summation operation. Following this aggregation step, we employ a Recurrent Neural Network (RNN) to update the hidden states associated with the links using the aggregated sum. In our investigation, we perform a total of 4 message passing steps. After completing these 4 steps, the newly updated link states are once again

subjected to an element-wise summation and then transmitted to the DNN. The role of the DNN is to predict the Q-value, serving as an estimation of the score for action a originating from the current state s.

3.4.3 Learning Algorithm

The core of our learning algorithm lies in the interaction between the DRL agent and the environment. At its heart, our approach relies on deep Q-learning [53] to estimate the Q-value associated with each state-action pair. A pseudo-code illustration is provided in Algorithm 1 to elucidate the learning process. We start by initializing some key parameters (lines 1-6). The training unfolds across a predefined number of episodes, where each episode corresponds to a distinct failure event. Within each episode, we identify the set of demands within the network that are affected by the failure event, $numberoffaileddemands$. We establish a replay memory buffer M, designed to store the actions executed by the DRL agent. This buffer serves as a pivotal guide in the learning process.

We commence by setting the cumulative *reward* to '0' and pre-computing k shortest paths. The *environment* is initialized (line 7). The environment generates a traffic demand, represented as a tuple *src*, *dest*, a state s and network spectrum *spec*. The *spec* is a matrix of the network links by number of wavelengths. It shows the wavelength occupancy of the network. The links associated with the failure event have their wavelength feature adjusted by setting all slots to '1'. It is important to note that the wavelength slots occupied by the affected demand on the active links remain unchanged, under the assumption that the failed demand will return to its original path once the failure is rectified.

Upon completing the initialization process, we enter a while loop (lines 9-19), which

continues until all the affected demands have been attempted for restoration on the network. For each impacted demand, Q-values are calculated for every one of the 40 path-wavelength combinations (lines 9-12). With Q-values computed for each state-action pair, the next action a is determined through the utilization of an $\epsilon - greedy$ exploration strategy (lines 15-16). As previously mentioned, the agent's options include selecting one of the K shortest paths with an available wavelength or rejecting the request. The chosen action is then applied to the environment, leading to the computation of a reward and the generation of a new state s (line 16). The number of impacted demands is not the same for every failure instance. We therefore compute the cumulative reward for each episode as the percentage of the failed demands that was successfully restored (i.e. $TotalRewards/ImpactedDemands$). Crucially, we retain the information regarding this state transition within the buffer (line 17), facilitating the training of the GNN.

3.5 Experimental Results

3.5.1 Setup

Our solution is implemented in Python 3.9 using Tensorflow [57] and evaluated using OpenAI Gym framework [58]. Training was carried out on core i7 computer with 32GB memory. In the simulation environment, the DRL agent is tasked with assigning a path and wavelength to a demand. It receives an immediate reward of '1' when the assignment is successful; otherwise, the reward is '0'. Success is defined as the demand being allocated a path where all the links have contiguous wavelength slots. Episodes conclude when a demand allocation fails to meet this criterion.

A series of experiments were conducted to determine the optimal optimization algorithm and hyperparameter settings. For the GNN model, feature vectors of dimension-

Algorithm 1 Pseudocode for DRL agent operation

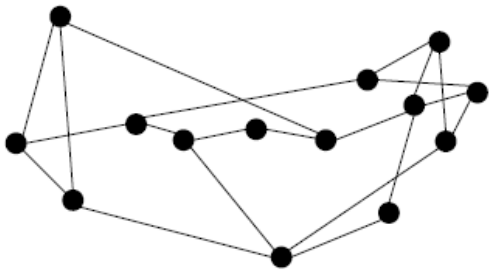
$E_s \leftarrow numberoffaileddemands$ ▷ Size of the episode
$k \leftarrow 4$ ▷ Number of shortest paths
$W \leftarrow wavelengthsAvailable$ ▷ Number of shortest paths
$M \leftarrow \{\}$ ▷ Initialize memory to store transitions
$done \leftarrow False$
$reward \leftarrow 0$
$s, src, dst, spc \leftarrow initializeEnvironment$
while $notfinished$ **do**
    $A\_qvalues \leftarrow \{\}$
    $A \leftarrow compute\_actions(src, dst, W, k)$ ▷ compute action space
    for $iin0, 1....size(A)$ **do**
        $A\_qvalues[i] \leftarrow compute\_qvalue(s, A[i])$
    end for
    $q\_value \leftarrow epsilon\_greedy(A\_qvalues)$
    $a \leftarrow get\_action(q\_value, A)$
    $r, done, s', src', dst', spc' \leftarrow env.step(s, a)$
    $M.add\_sample(s, src, dst, spc, r, s', src', dst', spc')$
    $reward \leftarrow reward + (r/E_s)$ ▷ episode reward is the ratio of restored demands to total demand requests
    $s \leftarrow s'$;$src \leftarrow src'$;$dst \leftarrow dst'$;$spc \leftarrow spc'$
end while

ality 64 were defined, incorporating the previously mentioned features. The size of the hidden states in the GNN is crucial as it determines the capacity to encode information. In scenarios involving larger network topologies and complex optimization tasks, larger hidden state vector sizes may be necessary.

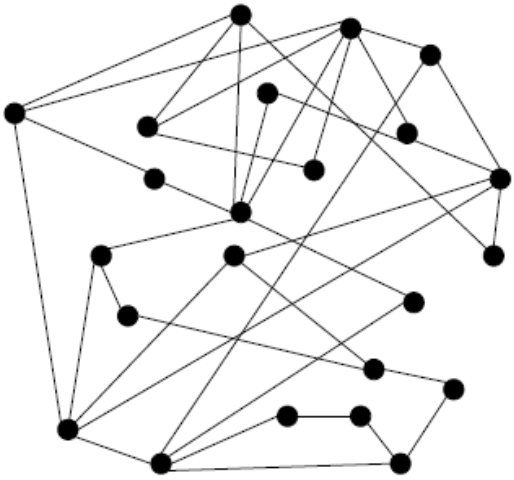
During each forward propagation of the GNN, a total of $M = 4$ message passing steps are executed using batch sizes of 32 samples. Four message-passing iterations in the GNN were selected based on standard practice and preliminary experiments that showed diminishing returns on representation quality improvement beyond four iterations, thus providing efficient inference with high-quality network embeddings. The optimization process employs the Adam optimizer [59] with a learning rate of 0.0001. The learning rate was determined experimentally as the best compromise between stable training and reasonable training convergence speed consistent with commonly accepted practices in DRL training. The exploration strategy follows an $\epsilon - greedy$ approach, commencing with an initial ϵ value of 1.0, which is maintained for the first 60 training iterations. Subsequently, ϵ exponentially decays at the start of each episode. An experience buffer with a capacity of 5,000 samples is implemented as a FIFO queue which provided adequate room for storing diverse training samples whiles limiting computational overhead. Regularization techniques, specifically $l2$ regularization, and dropout with a coefficient of 0.1 are applied to the readout function. The discount factor for future rewards is set to 0.95 to account for delayed rewards in the reinforcement learning framework.

3.5.2 Methodology

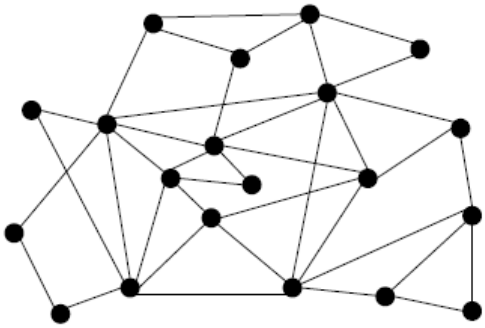
For each network configuration, we test with different numbers of demand requests randomly selected for all possible node to node demands in the network. For all four networks, we test with 50, 60, 70 and 80 demand requests. We compare our solution with



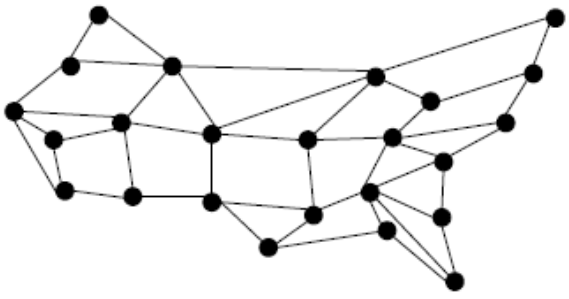
(a) 14-node NSFNET Network



(b) 24-node GEANT Network.



(c) 20-node EON Network.



(d) 24-node USA Network.

Figure 3.2: Test Networks

proactive restoration ($Pro-R$) and reactive restoration ($R-R$) approaches discussed in the literature.

In $Pro-R$, we define and reserve resources for one backup path designing the network to be 1-link failure survivable. During steady state routing, the proactive restoration approach only admits a demand request if it can reserve resource for a backup path that is link-disjoint for the primary path.

In $R-R$, we route the services using the shortest path as the primary route and first fit algorithm to assign wavelength. Additionally, four next shortest paths are pre-computed for restoration. However, resources are not reserved for the pre-computed back up paths. During failure, the backup paths are examined one after another from the shortest until the path with available resource to restore the impacted service. If none of the backup routes have spectrum, the restoration request is rejected. During restoration, the failed demands are recovered sequentially. For illustrative purposes, we select four real-world network topologies: 14-node NSFNET [40], 23-node GEANT[60], 20-node EON and 24-node USA network. The characteristics of the test networks are outlined in Table 3.2. We trained our DRL agent to choose the pre-computed route to restore a failed demand based on experiences. The agent learning from experiences is able to choose the best route for restoration that will improve the overall recovery rate of the network. We trained different agents for each of the 4 test networks. We trained for 3000 iterations and chose the model with the highest performance.

To evaluate the performance of the 3 solutions, we use two performance metrics of blocking probability and restoration probability and also presented computational complexity and runtime analysis. The blocking probability of a demand is estimated by the ratio of the number of blocked demands to the total number of demand requests during the steady state routing. Restoration probability is determined by the ratio of

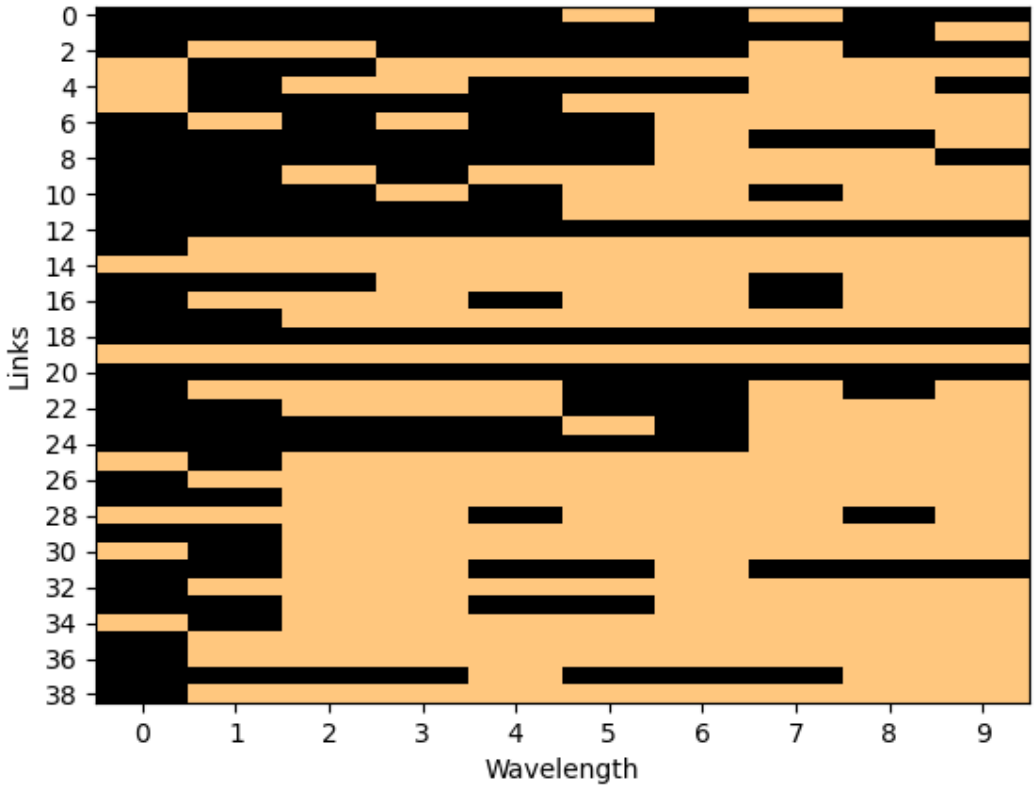


Figure 3.3: Wavelength Map for EON during steady state routing

the number of successfully restored demands to the original number of failed demands per failure instance. We determine the average restoration probability over multiple failure instances. In our experiment, a failure corresponds to single link failure. We also considered the following assumptions in our simulation:

- We assume our network demands to be permanent. This is because in typical backbone optical networks, network traffic is quite steady and consistent.

- During failure, wavelengths occupied by impacted demands on healthy links are not released. We assume that demands retain their primary path and will be rerouted when failure is fixed. However, we do not consider the post-failure state in our

Table 3.2: Characteristics of Test Networks

Topology	Nodes Size	Number of links	Average Node Deg
GEANT	24	37	3.1
NSFNET	14	21	3.0
EON	20	39	3.6
USA	24	44	3.7

simulation.

3.5.3 Performance Evaluation

Figure 3.4 shows the blocking probability of the 3 methods over multiple demand requests. During steady state routing, $Pro-R$ admits demands if there is enough resources to route both the primary path and the disjoint backup path. It then reserves resources on the backup path for demand recovery in the case of link failure. $R-R$ and our scheme employ the same strategy for steady state routing. A demand request is admitted if there are resources available to route the primary path. Shortest path and first fit algorithm are used to route the demand requests as seen in Figure 3.3 which depicts the wavelength map after routing 60 demands on the EON network during steady state. This graph represents a wavelength allocation matrix for a WDM illustrating spectrum utilization across various network links. The y-axis indicates the network links (numbered 0 to 38), while the x-axis shows the available wavelengths or spectrum slots (numbered 0 to 9). Orange or beige areas represent unused spectrum slots, indicating availability for future allocation, whereas black areas signify allocated wavelengths, showing active utilization for data transmission. The results in all four networks show that $Pro-R$ has high blocking rate. This is because reserving resources for restoration consume network

capacity and prevents routing of more demands. On an average, our method and $R-R$ perform 43% better than $Pro-R$ in the GEANT network, 54% better in the NSFNET network, 30% better in EON network and 34% better in USA network.

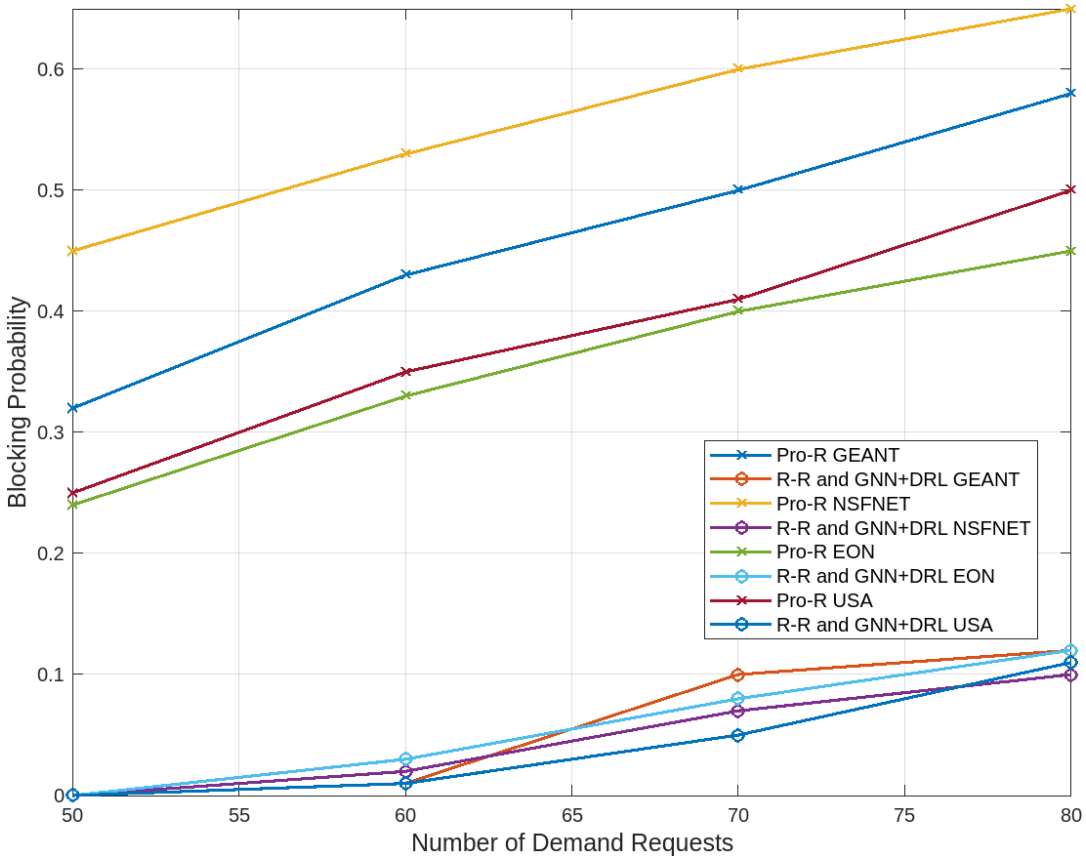
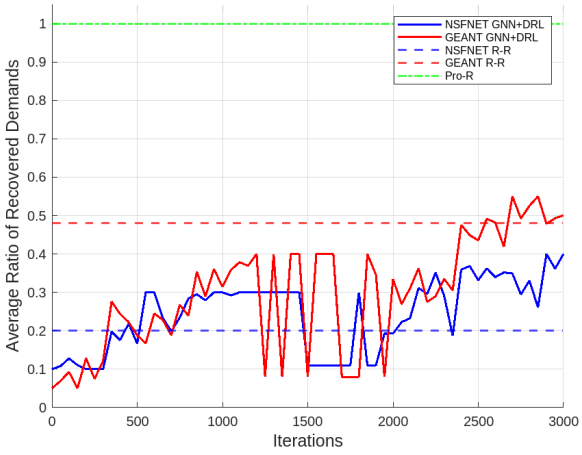
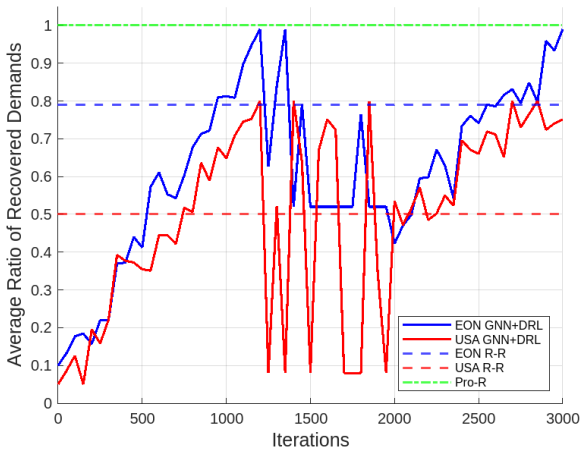


Figure 3.4: Blocking Probability

We also present the learning curves for the agents during training in Figure 3.5a and Figure 3.5b. An agent is trained for each topology. These curves show how the average restoration probability evolves during the training. We also show the restoration probability of the baseline methods. The baseline performances remain constant since no training is carried out for them. We observe that the agent is able to achieve better performances than the baselines when trying to maximize the average restoration prob-



(a) NSFNET AND GEANT results for 80 demand requests.



(b) EON AND USA results for 80 demand requests

Figure 3.5: Evaluation of the GNN+DRL agent during training across the four topologies. Each solid curve shows the average ratio of recovered demands achieved by the GNN+DRL agent versus the training iteration for the topology indicated in the legend; the dashed lines mark the corresponding constant R-R baselines, and the green dash-dot line marks Pro-R, which recovers all demands.

ability during failure. All the requests admitted in the proactive restoration scheme can be restored under single link faults. $Pro - R$ guarantees 100% restoration. Although our solution does not guarantee 100% restoration probability under the experimented network conditions, the figures clearly show that our scheme improves restoration probability significantly without network resource reservation. Figure 3.5a shows that during single link failures, our scheme has 40% restoration when 80 demand requests were admitted during the steady state routing on the $NSFNET$ topology. The is 50% better than the $R-R$ solution. Also in Figure 3.5a, when the $GEANT$ network has 80 demand requests, our scheme achieves 55% restoration probability, 14.5% better than $R - R$. Figure 3.5b show a similar trend, EON network achieving 14% better for 80 demands and USA network getting 28.4% better for 80 demands. Network survivability is very important parameter in network operation hence any improvement is deemed very significant. Figure 3.6 shows the restoration probability of the 3 methods over multiple

demand requests. Though $Pro - R$ achieves 100% restoration probability over all demand requests, it must be noted that it admits less number of demand requests given same resources as the other 2 methods. The figures show that our scheme always result in a significant improvement in restoration probability for different demand requests. For example, in the $GEANT$ network, for steady state loading of 50%, our scheme achieves 92.4% restoration probability, 11.8% better than $R - R$.

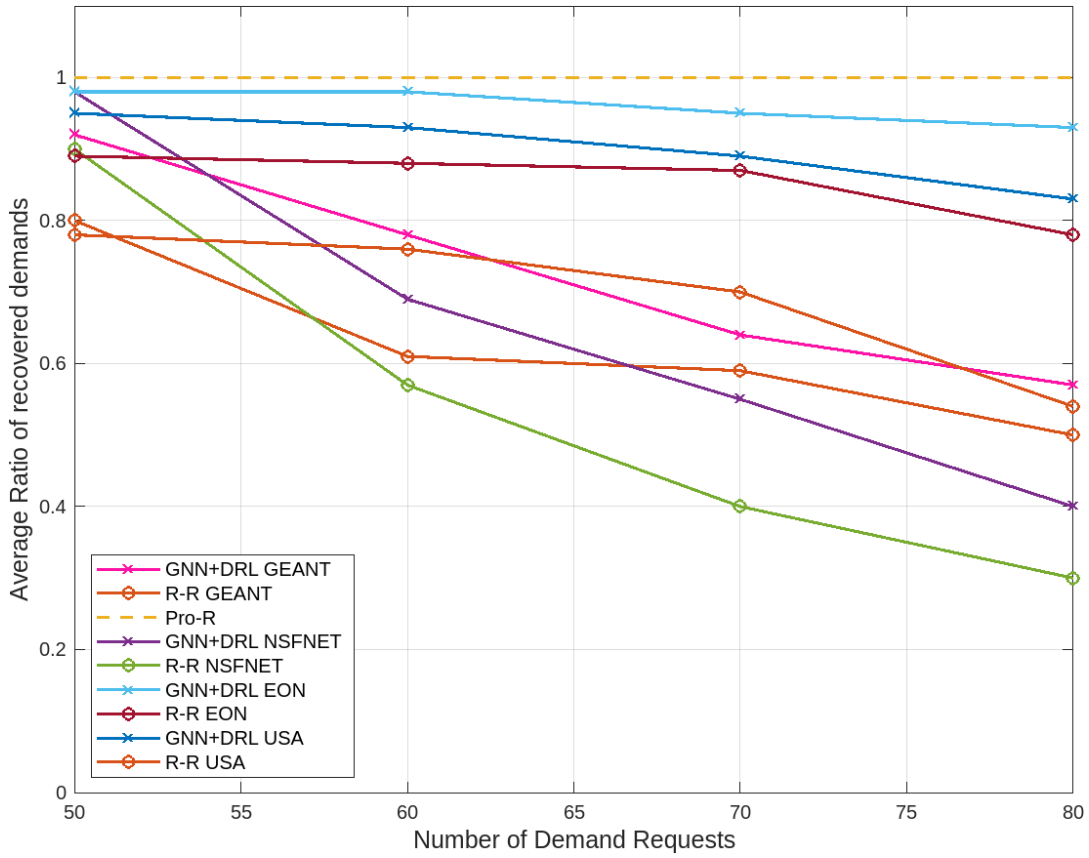


Figure 3.6: Restoration probability over different demand requests

3.5.4 Computational Complexity Analysis

To ensure a fair comparison between our proposed DRL+GNN solution and compared methods (Pro-R and R-R), we include an analysis of the computational complexity of

each approach. Our DRL+GNN solution complexity arises from two primary components: the Graph Neural Network (GNN) and the Deep Q-Learning (DQN) model. The GNN computational complexity is influenced by its message-passing framework, where each iteration involves aggregating information across the graph edges. The complexity of this process can be described as $O(M \times E \times F)$, where M is the number of message-passing iterations, E is the number of edges (links) in the WDM network, and F is the feature dimension of the input vector representing the network state. In this work, we set $M = 4$ iterations.

Additionally, the DQN model contributes to the overall complexity. The DQN evaluates the Q-values for each action, where the number of actions A is determined by the number of pre-computed paths (k) and available wavelengths (N). Therefore, the complexity of the DQN evaluation per state is proportional to $O(A)$, with $A = kN$, plus one for the rejection action, giving a total action space of $kN + 1$. Together, the GNN and DQN components give our DRL+GNN approach an overall complexity of $O(M \times E \times F + A)$. While this includes message passing and Q-value computation, both components scale efficiently with network size and are well-suited for dynamic WDM networks, particularly during the inference phase where real-time restoration decisions need to be made.

On the other hand, Proactive Restoration (Pro-R), a well-known benchmark method, involves pre-computing and reserving backup paths for all possible failure scenarios. The complexity of Pro-R primarily arises from the necessity to compute and maintain K-disjoint paths for each demand across the network, where K is the number of backup paths and V is the number of nodes. This leads to a time complexity of $O(K \times E \times V)$. Although this guarantees a quick restoration during failure, the upfront cost of resource reservation and path computation can be very high, particularly in large networks with

large demand requests.

In contrast, Reactive Restoration (R-R) adopts a more dynamic approach, computing alternate paths only after a failure occurs. The R-R method involves pre-computing multiple backup paths but without reserving network resources in advance, reducing upfront complexity. However, when a failure occurs, the system searches for available resources on the pre-computed paths, leading to a time complexity of $O(E \times V)$ for each restoration attempt. This can result in longer restoration times in large networks, especially when multiple paths need to be checked sequentially to find one with available resources.

To further highlight the differences, our DRL+GNN solution offers significant advantages in real-time restoration due to its ability to make decisions based on the pre-trained model. Although the training phase of DRL+GNN is computationally more intensive due to the iterative learning process (requiring 3000 episodes to train), the inference phase during restoration is efficient, as the trained agent can quickly select the optimal path-wavelength pair using the learned Q-values. This makes the proposed solution particularly suitable for real-time network restoration where speed is critical. In contrast, Pro-R has high complexity during the pre-computation phase due to resource reservation, while R-R incurs higher complexity during the restoration phase due to dynamic path search, leading to increased restoration times in large-scale networks.

3.5.5 Runtime Analysis

To evaluate the practicality of the proposed DRL+GNN solution, we performed a detailed runtime analysis to compare the computational efficiency of the Proactive Restoration (Pro-R), Reactive Restoration (R-R), and DRL+GNN method. The analysis includes two key metrics: the training time required for the DRL+GNN model and the decision

time (inference time) for all methods during failure restoration.

- **Training Time**: The DRL+GNN model was trained over 3000 iterations on all the topologies. The total training time for the NSFNET topology was approximately 2 hours, 3.5 hours for GEANT and EON and USA took about 5 hours. While the training process incurs a significant upfront computational cost, the model's generalization ability allows it to perform efficiently during the real-time restoration phase.

- **Decision Time (Inference Time)**: Once trained, the DRL+GNN model was evaluated based on the time required to determine restoration for a single failed demand, comparing it to Pro-R and R-R methods. We measured the average decision time (in milliseconds) over multiple failure instances.

Table 3.3: Runtime Analysis Results

Method	Training time(hrs)	Average Decision time(ms)
Pro-R	N/A	0.05
R-R	N/A	20
GNN+DRL	2 (NSFNET), 3.5 (GEANT,EON),5 (USA)	0.12

The runtime analysis highlights the trade-offs between pre-computation, dynamic restoration, and real-time decision-making in different methods. While Pro-R offers the fastest decision time due to its pre-computation phase, it suffers from higher resource utilization and limited adaptability to unforeseen failures. The R-R approach, though resource-efficient, incurs significant delays during the restoration process due to its dynamic search for viable paths.

In contrast, the DRL+GNN solution, once trained, strikes a balance between computational efficiency and decision speed. Although the training phase introduces higher upfront computational costs, the ability to generalize to unseen network topologies ensures fast and reliable restoration during the inference phase, making it particularly suited for large-scale WDM networks. Given the increasing complexity of modern optical networks, the efficient inference time of DRL+GNN ensures that it can operate in real-time restoration scenarios without the need for extensive pre-computation or resource reservation.

It is important to note that the higher upfront training time for the DRL+GNN model, while significant (2–5 hours, depending on topology), does not negatively impact its practical utility in real-world WDM network operations. Training the model is an offline, one-time procedure per network topology, conducted well before actual deployment. Conversely, during real-time network failures, rapid inference time directly translates to quicker restoration and minimal network downtime, metrics that are critical in operational networks. Hence, the decision-making time is the key criterion when assessing the practicality of restoration solutions. In this context, the slightly longer offline training time of the DRL+GNN model is fully justified by the substantial gains in inference-time efficiency, adaptability to dynamic network states, and significantly enhanced restoration performance, making it suitable for real-world deployment scenarios.

3.5.6 Discussion

In addition to presenting the numerical results, it is crucial to explore why our proposed DRL+GNN model outperforms the benchmark methods, Proactive Restoration (Pro-R) and Reactive Restoration (R-R). Several key factors contribute to the superior performance of the DRL+GNN model, particularly its ability to dynamically adapt, optimize

resource allocation, and make real-time decisions, all of which give it a distinct advantage over traditional methods. Also, we provide specific considerations of DRL and GNN in WDM network restoration.

3.5.6.1 Adaptive Learning and Generalization Capabilities of DRL+GNN

One of the primary reasons for the improved performance of the DRL+GNN model is its ability to adapt dynamically to network conditions and generalize effectively to new failure scenarios. Unlike Pro-R, which relies on pre-computed, static restoration paths, and R-R, which reacts to failures by searching for paths only after the failure occurs, the DRL+GNN agent learns from a variety of network states and failure situations during the training phase. This allows the model to develop a robust decision-making strategy that can be applied to unseen failure conditions. Through the training process, the DRL agent encounters numerous network states and failure scenarios, learning to make optimal decisions based on historical experiences. The Graph Neural Network (GNN) component allows the agent to encode the state of the network in each episode, improving the agent's ability to assess the impact of failures and dynamically select the most efficient restoration paths. This adaptive learning capability enables the DRL+GNN model to outperform Pro-R and R-R, particularly in scenarios where demand patterns and failure conditions are unpredictable. The agent does not rely on static pre-computed paths (as in Pro-R), but instead adapts its decision-making policy based on real-time conditions, which contributes to faster and more efficient restoration.

3.5.6.2 Efficiency in Utilizing Network Resources

The DRL+GNN model performs well at optimizing resource allocation across the network, which is a key factor in its performance advantage over the benchmark schemes.

The model efficiently balances the use of wavelengths and paths, ensuring that network resources are used optimally without unnecessary reservation or allocation delays. The Pro-R approach pre-allocates backup resources by reserving alternative paths in advance for potential failures. While this method can ensure quick restoration in some cases, it often leads to under-utilization of network resources. Reserved resources may remain unused, resulting in inefficiency and increased blocking of new demands due to the limited availability of unreserved paths. The R-R method, on the other hand, allocates resources only after a failure has occurred. This prevents under-utilization of the resources but does not optimize the use of the resources during restoration. In contrast, the DRL+GNN agent continuously learns to make resource allocation decisions in real-time, without the need for pre-reservation. The GNN component helps the agent understand the current resource availability and link states across the network, while the DRL agent decides on the best restoration path-wavelength pair by optimizing for minimal contention and maximal resource utilization. This dynamic decision-making process minimizes the blocking probability while ensuring efficient use of network resources, particularly in scenarios with heavy traffic or frequent failures.

3.5.6.3 Reduced Restoration Time

A critical factor in network restoration is the speed of recovery after a failure occurs. Restoration time directly impacts the availability of services and the user experience, especially in time-sensitive applications. Pro-R benefits from having pre-reserved backup paths, allowing it to restore services relatively quickly. However, since these backup paths are pre-determined and static, they may not always be the most optimal paths under varying network conditions. This can lead to suboptimal path utilization and potential delays in cases where reserved paths are congested or unavailable. R-R, by contrast,

incurs longer delays due to the need to perform real-time searches for alternative paths during a failure. This process can become time-consuming, especially in large networks with multiple failures or under high-demand conditions. The DRL+GNN model leverages its pre-trained policy to make real-time restoration decisions with minimal delay. The GNN's ability to model the entire network topology and capture the current state of the links ensures that the DRL agent has an accurate understanding of the network environment, allowing it to quickly select the most efficient restoration path. As a result, the DRL+GNN model achieves faster restoration times compared to R-R and matches closely to Pro-R.

3.5.6.4 Scalability and Robustness to Network Changes

Another key factor contributing to the superior performance of the DRL+GNN model is its inherent scalability and robustness to changes in network topology and demand patterns. Unlike the Pro-R and R-R methods, which struggle to maintain efficiency in large-scale or highly dynamic networks, the DRL+GNN model can scale effectively to large network topologies while maintaining high performance. The GNN architecture used in the model allows for efficient processing of large network graphs, as the agent is able to model the relationships between nodes and links regardless of the size or complexity of the network. This enables the DRL+GNN model to scale to networks with thousands of nodes and links without suffering from the performance degradation commonly seen in Pro-R and R-R methods. Additionally, the DRL+GNN model is more robust to dynamic changes in the network, such as multiple simultaneous failures or sudden changes in traffic patterns.

3.5.6.5 Handling Multiple Failure Scenarios

WDM networks often face not just single link failures but also multiple simultaneous failures, which can severely impact the performance of restoration mechanisms. Traditional methods like Pro-R and R-R are generally optimized for handling single failure events and may not perform efficiently when confronted with multiple concurrent failures. The DRL+GNN model, however, has the capability of generalization and hence can adapt to multiple failure scenarios. As a result, the DRL+GNN model demonstrates superior performance in restoring demands even when faced with multiple failure events, something Pro-R and R-R struggle with.

While the DRL+GNN model demonstrates superior performance in restoring WDM networks compared to Pro-R and R-R, there are areas where the model can be enhanced for even greater effectiveness. In particular, scalability to large-scale networks, training efficiency, adaptability to highly dynamic conditions, and energy efficiency are areas that present opportunities for improvement.

Potential enhancements to the model include:

- Enhancing Scalability by adopting hierarchical learning or graph sampling. The model could scale more efficiently to very large networks without significantly increasing complexity.

- Improving Training Efficiency using Transfer learning and parallel training techniques. This could reduce the time required to train the DRL agent, enabling faster deployment in new network environments.

- Adaptability to dynamic conditions by incorporating meta-reinforcement learning and proactive failure prediction. This will aid the ability to generalize across diverse network scenarios, making it more robust to dynamic traffic patterns and multiple

failures.

- Enhancing the model to handle node failures explicitly to improve its robustness in real-world environments where both link and node failures can occur simultaneously.

3.5.6.6 Specific Considerations of DRL and GNN in WDM Network Restoration

While DRL and GNNs have been effectively applied in various network scenarios such as routing optimization in Software-Defined Networks (SDNs), topology optimization, and QoS-aware routing in Mobile Ad-hoc Networks (MANETs), the application of these methods to WDM Network Restoration poses unique challenges and requirements distinctly different from these other scenarios:

1. **Strict Real-time Requirements**: Unlike general network routing or topology optimization tasks that may permit longer decision times, WDM restoration demands extremely rapid response (ms) following network disruptions to minimize downtime and service interruptions. Consequently, the design of DRL and GNN models for WDM restoration specifically emphasizes low-latency inference, rapid state encoding via GNN message-passing, and streamlined DRL policy decisions, aspects not necessarily prioritized in other network applications.

2. **Wavelength Continuity**: WDM restoration uniquely involves strict wavelength continuity constraints, meaning that selected restoration paths must allocate consistent wavelength resources along the entire path. This adds a significant layer of combinatorial complexity not usually present in other network scenarios, e.g., SDN or MANET routing, requiring specialized action-space definition, feature represen-

tation, and policy learning within the DRL framework, enhanced specifically by GNN's topological insights.

3. **Highly Dynamic Failure Patterns**: Restoration scenarios in WDM networks frequently involve multiple and simultaneous fiber cuts or node failures due to natural disasters, human error, or equipment malfunctions. These failures create dynamic and unpredictable conditions not typically encountered in conventional network routing or static optimization tasks. Hence, DRL agents for WDM restoration must robustly learn adaptive, dynamic policies capable of quickly responding to new, unforeseen network states, a capability possible because of the generalized representations learned by GNNs.

4. **Scalability and Generalization Across Diverse Topologies**: Unlike many scenarios where networks remain relatively static or where each topology is specifically trained, WDM restoration often requires robust generalization from learned experiences to entirely new and unseen network topologies. The DRL+GNN framework explicitly addresses this scenario-specific challenge by leveraging GNN's inductive biases that inherently enable generalization across arbitrary network structures, thus significantly outperforming approaches used in less topologically diverse scenarios.

5. **Dual Proactive-Reactive Restoration Capability**: Traditional DRL/GNN methods typically handle either proactive provisioning (pre-allocation) or reactive rerouting separately. In contrast, effective WDM network restoration demands a unified framework capable of simultaneously optimizing pre-computed restoration paths (proactive) and dynamically rerouting paths post-failure (reactive). Our hybrid DRL+GNN method uniquely integrates these two modes of operation within

a single, coherent learning and inference framework—distinctly tailored for WDM network characteristics.

By explicitly identifying and addressing these scenario-specific differences, our DRL+GNN approach is uniquely positioned and tailored to meet the specialized demands of WDM network restoration, clearly distinguishing our approach from methods previously applied to other networking contexts.

3.6 Conclusion

Network survivability is a major and critical concern in the design and operation of WDM network. WDM networks are subjected to multiple network interruptions hence the need for quick and reliable restoration mechanisms. In this chapter, we presented a novel approach for enhancing network survivability in WDM networks by integrating DRL with GNN. This DRL+GNN framework addresses both proactive and reactive restoration scenarios by intelligently selecting restoration paths and wavelengths based on real-time network conditions. Our model's ability to generalize across varying network topologies and dynamically adapt to link failures results in superior restoration performance compared to traditional Proactive Restoration (Pro-R) and Reactive Restoration (R-R) schemes.

The experimental results demonstrated the significant advantages of the proposed DRL+GNN solution. Our method outperformed the benchmark approaches across various network topologies (NSFNET, GEANT, EON, USA) in terms of both restoration probability and blocking probability. Specifically, the DRL+GNN model achieved up to 28.4% better restoration probability compared to R-R, while avoiding the resource inefficiencies associated with Pro-R. Moreover, the runtime analysis revealed that, once trained, the DRL+GNN model offers a compelling balance between decision time and

resource allocation, making it suitable for real-time applications in large-scale WDM networks.

The scalability, adaptability, and resource optimization capabilities of the DRL+GNN model make it a promising solution for future optical network management, particularly in scenarios involving dynamic traffic patterns and multiple failures. While the training phase incurs higher computational costs, the model's ability to generalize across unseen topologies ensures long-term efficiency during the inference phase.

In conclusion, the integration of DRL and GNN offers an effective approach for addressing the challenges of WDM network restoration, providing a scalable, efficient, and adaptive solution that significantly improves network survivability.

Beyond the specific restoration results obtained in WDM networks, this chapter establishes an important thesis-level insight: graph-based learning is particularly effective when network recovery decisions depend on topology, resource state, and interdependent routing constraints. At the same time, the chapter also shows that graph representation alone is not sufficient for this class of problem; because restoration is inherently sequential and decision-oriented, it must be coupled with reinforcement learning to translate structural understanding into adaptive control. In this way, the chapter supports the broader claim of the thesis that GNNs form a unifying representational backbone for intelligent network management, while task-specific auxiliary methods are required depending on the operational nature of the problem.

Chapter 4

Graph Neural Network-Based Radio Link Failure Prediction in 5G Networks

This chapter addresses the second network management setting in the thesis: *predictive management*. Rather than reacting after disruption has occurred, the focus here is on anticipating radio link failures in 5G networks before service degradation becomes critical. Within the broader dissertation, this chapter demonstrates how graph-based learning can improve proactive reliability management by capturing spatial dependencies among radio links, environmental influences, and operational context that are difficult to model using independent-link prediction approaches.

4.1 Introduction

The advent of 5G technology, characterized by high data transfer speeds, extensive capacity, and low latency, addresses the growing demand for high-speed Internet access, nec-

essary for applications like autonomous vehicles, IoT networks, and high-speed gaming. These applications require robust communication and maintaining network connectivity is key. 5G networks use millimeter-wave spectrum (24-40GHz) for the Radio Link (RL) [61] which travel short distances and can be affected by weather conditions. It is expected that more future network infrastructure will be built using these high frequency bands [61]. Most network operators invest hugely in monitoring these networks to respond quickly to RLF, but this reactive approach can lead to significant monetary losses from downtime. One of the primary challenges in preventing RLFs is the unpredictable nature of failures, which arise from environmental conditions, infrastructure configurations, and interference patterns. While traditional approaches rely on rule-based monitoring systems, they fail to generalize across dynamic network conditions. This necessitates a proactive, predictive approach that leverages machine learning (ML) to forecast failures before they occur, enabling preventive actions. However, traditional ML models struggle to predict RLFs effectively due to the highly imbalanced nature of failure events, which occur in a small percentage of network operations and the complex dependencies between radio link performance and environmental conditions. In contrast, Graph Neural Networks (GNNs) offer a more powerful alternative by modeling spatial dependencies among radio links, capturing relationships between geographically distributed network nodes, and enhancing predictive accuracy through adaptive learning mechanisms.

GNNs [62] are a novel machine learning architecture best suited for graph-based data. This research uses GNNs to model radio link failure prediction by analyzing historical weather data, KPIs, and configuration data. Our model leverages a live network dataset that is highly imbalanced, presenting a significant challenge. By using two days of historical weather data, we predict whether a link fails or not. We validate our model with k-fold cross-validation and compare it against traditional models like LR, SVM,

and LSTM networks. While traditional machine learning models struggle to capture the complex spatial dependencies in radio link networks and often require extensive preprocessing, GNNs excel at learning from graph-structured data, making them better suited for this problem. They effectively model relationships between links and external factors such as weather, enabling more accurate predictions. To our knowledge, this is among the early studies to apply GAT for RLF prediction, utilizing attention mechanisms to focus on the most significant features.

Practical adoption hinges on the following two challenges:

- operationalizing predictions with explicit latency/availability Service Level Objectives (SLOs) and actionable thresholds, and
- learning under extreme class imbalance without sacrificing minority recall.

We address both: our deployment blueprint details end-to-end latency budgets, redundancy, and Network Operations Centers (NOC)/cloud/edge placements, while ablations quantify how focal/SMOTE-family methods affect Precision-Recall (PR)-Area Under the ROC Curve (AUC), precision-recall trade-offs, and alert volume. Together, these elements convert GNN accuracy into operator-usable early warnings, aligning model behavior with ticketing, dashboards, and closed-loop mitigation.

The contributions of this chapter include:

- A graph formulation for radio-link failure prediction that integrates radio KPIs, site attributes and short-horizon historical weather context.
- A GAT predictor that leverages neighborhood information to improve rare-failure detection compared to non-graph baselines on a real operator dataset.
- A systematic imbalance study, including ablations and comparisons of multiple imbalance mitigation strategies, showing their impact on Precision/Recall, F1-score,

ROC–AUC, and PR–AUC under extreme skew.

- A deployment-oriented blueprint detailing end-to-end data flow, carrier-grade operational considerations such as latency and availability, threshold tuning for alert budgets, and NOC/cloud/edge placement options.

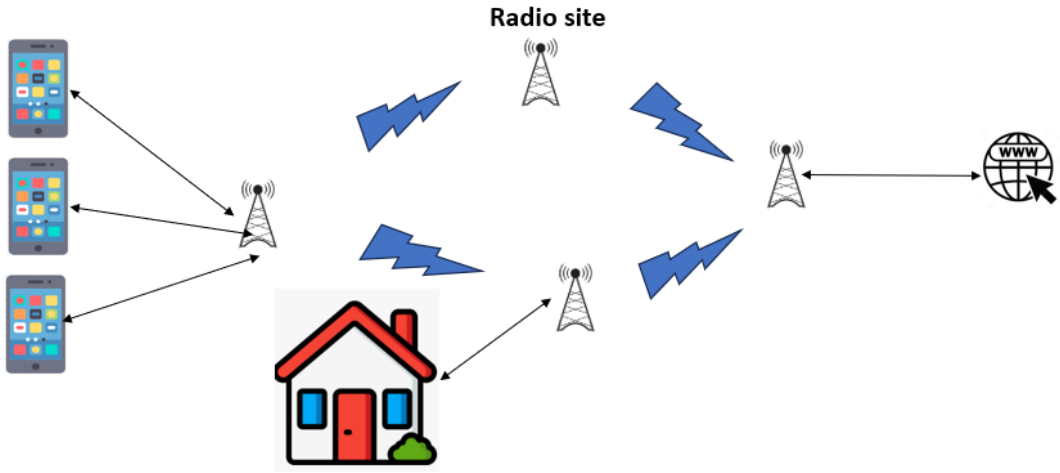


Figure 4.1: Radio site in a 5G network

4.2 Related Works

This section reviews the prior work most directly relevant to radio link failure prediction in 5G networks. It complements the cross-cutting survey in Chapter 2, focusing on the failure-prediction studies that motivate the graph-based approach developed in this chapter.

In this section, we first survey how weather conditions influence wireless links, then review learning-based approaches to radio link failure (RLF) prediction, including classical ML, deep learning, and graph-based methods for the RAN. We conclude with emerging directions in explainability and federated learning that affect practical deployment.

4.2.1 Impact of Weather on Wireless Networks

A body of work [63–66] establishes weather as a dominant exogenous factor in wireless degradation. For mmWave systems, [63] shows that precipitation is a primary driver of attenuation and can precipitate outages; the impact increases with path length and carrier frequency, effectively shrinking link budgets and eroding fade margins. Expanding beyond rainfall, [64] considers humidity, fog, and temperature inversions and highlights how absorption, scattering, and refractivity gradients exacerbate blockage and multipath variability. Surveys, such as [65], connect these effects to mitigation mechanisms (e.g., adaptive modulation/coding, beam re-steering, dynamic handover), but largely focus on reactive control rather than proactive failure anticipation. Finally, [66] analyzes seasonal and spatial heterogeneity in rain-rate and attenuation, motivating prediction models that incorporate fine-grained spatiotemporal weather dynamics rather than static thresholds. Collectively, these studies motivate RLF predictors that leverage weather context and account for spatial locality.

4.2.2 Learning-Based RLF Prediction: Classical and Sequential Models

Early and widely adopted RLF predictors use classical supervised learning on engineered features. On the same operator dataset considered here, [67] employs an ensemble of tree-based learners (RF, LGBM, XGB) driven by weather and KPI features, demonstrating the feasibility of ML-based RLF prediction in operational settings. However, the approach does not explicitly incorporate historical weather context and provides limited reproducibility details regarding preprocessing. Probabilistic approaches such as Bayesian Neural Networks (BNNs) have also been explored for weather-related failures [68], offering potential robustness and uncertainty awareness, but are typically validated

under less extreme imbalance than encountered in our dataset.

Sequence-based deep learning has been used to exploit temporal structure in KPIs and weather. For example, [69] introduces an LSTM-autoencoder formulation that leverages spatiotemporal correlation between radio links and weather forecasts, and [70] proposes an LSTM+ variant that uses a multi-day (10-day) historical weather window to predict next-day link state. These models illustrate the value of temporal context, yet they often remain site/link-centric: spatial dependencies among neighboring sites are not explicitly represented as a graph, and performance under extreme class imbalance is not always characterized using metrics such as PR–AUC.

4.2.3 Deep Learning for Wireless Reliability

Recent work has developed specialized deep architectures to capture non-linear interactions and extend prediction horizons. Govindasamy et al. [71] propose DARMMCNN, which integrates multiscale attention within a residual CNN pipeline and uses feature selection/noise filtering to improve robustness; they report prediction up to a 5-day horizon with only minor accuracy trade-offs relative to shorter-term predictors. Kazi et al. [72] propose GenTrap, a hybrid GNN–Transformer architecture that combines weather-context aggregation with long-range temporal modeling; they report strong results across datasets (e.g., F1-score of 0.93 on a rural dataset and 0.79 in a denser urban scenario), highlighting the benefit of jointly modeling spatial and temporal structure. Other hybrids such as CNN–LSTM [73] show benefits in proactive handover contexts, while testbed-driven studies [74] demonstrate high accuracy on controlled environments but may not reflect rare-event performance or operational false-alarm trade-offs in operator-scale deployments. Overall, deep models improve expressiveness and horizon length, but many still lack an explicit, interpretable mechanism for encoding shared exogenous exposure

among sites.

4.2.4 Graph Neural Networks in RANs

GNNs have emerged as a natural tool for RAN problems where entities and interactions form a graph. In scheduling and interference mitigation, [75] models links and interference edges with a GCN-based scheduler and reports improvements over greedy heuristics in throughput maximization. Beyond scheduling, GNNs have been used for multi-RAT modeling and optimization [76], and for fault/anomaly detection with spatial-temporal hybrids: "Simba" [77] combines graph aggregation with a Transformer to diagnose anomalies and root causes in 5G RANs, reporting strong F1-scores for several fault types. Graph-based learning has also been applied to mobility and handover optimization [78], where UE–cell graph modeling reduces ping-pong events and RLF rates compared to threshold-based policies. While these studies demonstrate the advantage of topology-aware learning, graph construction choices and deployment constraints (thresholding, alert budgets, and latency) are not always explicitly articulated in the RLF prediction setting.

4.2.5 Explainability and Federated Learning Trends

As models become more complex, two directions are increasingly important for operational adoption: explainability and federated learning. Explainability methods such as GNNExplainer [79] and its extensions [80, 81] aim to identify influential subgraphs and feature subsets driving predictions, supporting operator trust and actionable diagnostics. Recent wireless case studies report that explainable graph models can surface human-interpretable failure drivers (e.g., localized rainfall combined with KPI degradation) [82]. Federated learning addresses data residency and privacy constraints by enabling collab-

orative training without centralizing raw KPI/failure data. Federated GNN frameworks such as FedPerGNN [83] and related approaches [84, 85] demonstrate that decentralized training can approach centralized performance on graph tasks, though communication cost and graph heterogeneity remain practical challenges. These trends motivate RLF predictors that are not only accurate, but also deployable, explainable, and amenable to privacy-preserving training.

Table 4.1: Comparison of representative related works on weather-aware wireless reliability and RLF prediction.

Work	**Technique**	**Reported outcome**	**Strengths**	**Limitations**
[63, 64, 66]	Weather propagation studies	Identify key weather drivers and seasonality	Physical grounding; motivates weather features	Not predictive models
[67]	RF/LGBM/XGB ensemble	Feasibility on operator dataset	Strong classical baselines; efficient	Limited temporal history; limited preprocessing detail
[68]	Bayesian NN	Robustness to imbalance (reported)	Uncertainty-aware potential	Less severe imbalance; limited spatial modeling
[69, 70]	LSTM variants	Temporal gains using weather/KPI history	Exploits temporal context	Often site-centric; spatial relations not explicit
[71]	Attention CNN (DARMMCNN)	Multi-day horizon (up to 5 days)	Longer horizon; attention over time scales	No explicit graph relational modeling
[72]	GNN+Transformer (GenTrap)	Strong F1 across datasets	Joint spatial-temporal modeling	Complexity; deployment details vary
[77, 78]	GNNs for RAN faults/mobility	Improved fault detection / reduced RLF rates	Topology-aware learning	Different targets; graph design/deployment often under-specified
This work	Weather-footprint GAT + imbalance ablation + deployment blueprint	F1=0.717; PR–AUC emphasis; sub-5-min ingest-to-alert	Sparse interpretable graph; rare-event focus; operational framing	Depends on station coverage; offline evaluation

4.2.6 Research Gaps and How We Bridge Them.

The literature highlights three recurring gaps. First, many predictors are evaluated with metrics or class distributions that do not adequately reflect rare-event detection performance, and the operational precision–recall trade-off is often under-emphasized. Second, several approaches remain link/site-centric or rely on generic proximity graphs, limiting the ability to encode *shared exogenous exposure*—for example, localized weather footprints affecting multiple neighboring links. Third, deployment considerations such as ingest-to-alert latency, threshold tuning for alert budgets, and carrier-grade availability targets are frequently not described. Our work bridges these gaps by constructing sparse graphs that connect sites sharing the same k-nearest meteorological station sets (capturing shared weather measurement footprints), explicitly validating imbalance mitigation through ablations, and providing a deployment-oriented analysis that maps model outputs to operational constraints and tunable decision thresholds.

4.3 Dataset Description and Preprocessing

This section presents our proposed GNN based failure prediction solution. Our approach comprises multiple steps including data cleaning and preprocessing, GNN model design, training, and validation as well as model testing. In the preprocessing stage, we correlate the radio site and weather station data and merge. We then transform categorical data and perform data normalization as well as data balancing. We train the proposed model and three other machine learning models using the same data and test the efficiency of the models. We provide more details in the following sections.

4.3.1 Dataset

The dataset utilized in this study originates from Turkcell, a prominent service provider based in Turkey. The dataset consists of a collection of radio link performance metrics, configuration attributes, and weather observations from meteorological stations within same region spanning the years 2018 to 2020 [86]. To protect sensitive information, certain configuration and performance metrics have been categorized and normalized respectively while preserving data integrity. Additionally, the dataset incorporates a pairwise distance matrix for weather stations and radio sites without the precise location details for security considerations. The dataset has both categorical and numerical features. The dataset consists of 3 main categories explained below.

1. Radio link data: This data provides information on radio link sites (*rl-sites*) and radio link key performance indicators (*rl-pkis*).

    - ***rl-sites***: This data contains site specific information such as site id, height and clutter class. The site id is a unique identifier for the site. The height represents the distance above sea level and clutter class indicates the environment type (e.g. airport, dense tree area) where the site is located. The Clutter class is a categorical terrain label describing the dominant physical environment surrounding a site which is derived from the operator's planning/geospatial inventory. It captures macro-environmental factors that influence radio propagation and link reliability (e.g., obstruction, scattering, and multipath conditions), such as open/rural areas, suburban/urban density, vegetation/forest, water bodies, and special zones (e.g., airport/industrial corridors). In our dataset, there are 21 clutter categories, and we encode this attribute as a categorical feature (one-hot encoding) for model training. There are 1674 unique radio sites and 21 clutter classes.

- ***rl-kpis***: This data contains metrics used to assess the performance of the radio sites. There are 19 columns in the table comprising daily performance and configuration attributes recorded per radio link. Key columns include: *type*, *datetime*, *tip* (operator-specific link identifier), *mlid* (microwave link ID), *mw_connection_no* (microwave connection/circuit number), *site_id* (radio site identifier), *card_type* (radio/microwave card/hardware type), *adaptive_modulation* (indicator/configuration of adaptive modulation), and *freq_band* (operating frequency band). Performance counters include *severaly_error_second* (severely errored seconds within the measurement interval), *error_second* (errored seconds), *unavail_second* (unavailable seconds), *avail_time* (available time), and *bbe* (background block errors / errored blocks counter). Radio/physical-layer attributes include *rxlevmax* (maximum received signal level over the interval), *capacity* (configured/nominal link capacity), and *modulation* (modulation scheme in use). The binary label *rlf* indicates whether a radio link failure event occurred (1) or not (0) for the corresponding record. There are 1992986 samples in this data.

2. Weather station data: This data provides details on weather stations (met-stations), historical hourly weather realizations (met-real) and five-day weather forecasts information on any given day (met-forecast).

   - ***met-stations***:: Just like the ***rl-sites***, this data gives specific information on considered weather stations within the region. The details include station identifiers, height above sea level and clutter class. There are 20 unique sites and 8 clutter classes.
   - ***met-forecast***: This data shows weather forecast for upcoming five days. Some of the features include date-time, report-time, temperature, humidity,

wind directions etc. There are 479241 samples where each entry is uniquely identified by station number, observation date, and measurement hour.

- ***met-real***: This data provides hourly historical weather data. These are actual weather realizations. The features include temperature, maximum and minimum temperatures, wind speed, wind direction, etc. There are 16 features in total.

3. Distance Matrix: This provides a pairwise distance between weather station to weather station and weather station to radio station.

    - ***distances***: it is a matrix representing the distances between all sites in the dataset. Exact locations of sites are not provided for security reasons.

Table 4.2 provides a summary of all constituents of the used dataset.

Table 4.2: Summary of Dataset

Data	Number of Features	Samples
rl-sites	3	1674
rl-kpis	19	1992986
met-stations	3	20
met-real	16	479241
met-forecast	38	20323
distances	1694	1694

4.3.2 Data Preprocessing

The data set used is highly imbalanced. The data set has about 0.061% failure cases. Due to this, exhaustive preprocessing steps are adopted to prepare the data for the prediction

of radio link failure. The purpose of this research is to predict radio link failure using current and historical weather conditions. We utilized the $rl-sites$, *rl-kpis*, *met-stations*, *met-real* and *distances* data from all the twelve months in 2019 for model training and validation.

Data Correlation and Aggregation. Firstly, the hourly data (*met-real*) is aggregated on daily basis keying on *station_no* and *datetime*. In our method, we aim to aggregate weather data from the k closest weather stations to each radio link site. However, some radio link sites may not have up to k weather stations within a reasonable proximity. This is due to the distance constraint applied by our algorithm (Algorithm 1), which determines the closeness of weather stations based on the calculated pairwise distances between radio link sites and weather stations. The algorithm selects the nearest weather stations up to a predefined distance threshold, beyond which stations are considered too far to have a significant impact. As a result, if fewer than k weather stations are within this distance threshold for a particular radio link site, the algorithm aggregates data from the maximum number of available weather stations. To predict link failure for any node on the day, we use the radio link features (KPI and site information) and the aggregated weather information from the k nearest weather stations for that day and the past 2 days. We use a two-day historical weather window as a practical trade-off between temporal context and data completeness. In our dataset, missing weather measurements are non-negligible, and the fraction of incomplete samples increases rapidly as longer historical windows are used; this either reduces the number of usable training instances or requires heavier imputation. Restricting the history to two days preserves more valid samples and limits imputation-induced noise, while still capturing near-term meteorological dynamics relevant to next-day RLF prediction. The following steps are taken to complete the data aggregation and correlation phase.

1. For each *rl-kpi* record, we extracted the corresponding aggregated weather stations features for the day. The significant weather features used are *temp, temp_max, temp_min, wind_dir, wind_speed, wind_dir_max, wind_speed_max* and *humidity*, a total of 8 features.

2. For each *site-id* and *mlid* pair, we appended the radio link predictions for the 2-previous days to match the *rl-kpi* record.

3. Also, we merged the features described in step 1 for the 2-past days to the *rl-kpi* record naming them as *temp_day1, temp_day2, wind_dir_day1* etc.

4. After combining the outlined features, the total number of features in the dataset sums to 45 (19 features from *rl-kpi* data, 24 features from *met-real* data representing weather information for current day and 2-previous days and 2 features for the 2-previous days radio link predictions).

Finally, for each RL–RL pair, we computed the overlap between their k-closest weather-station sets (with $k = 3$ in all experiments). The information is used in modelling our graph neural network and will be discussed later.

Distance-threshold selection. To ensure that every RL site is associated with at least one meteorological station, we derive a global association threshold from the geometry of the dataset described in Algorithm 1. For each RL site r_l, we compute the distance to its closest weather station. This yields a set of nearest-station distances $\{d_1, \ldots, d_R\}$ for R RL sites. We then define the WS–RL association threshold as the maximum nearest-station distance across all RL sites. By construction, this choice guarantees that every RL site has at least one weather station within d_{th}, avoiding isolated nodes due to sparse station coverage and reducing missing weather assignments.

Handling null values. Based on the nature of the dataset, deleting every record

Algorithm 2 Algorithm for determining the WS–RL association distance threshold

Input: Set of all radio link sites
Output: Maximum threshold distance
$RL_s = [rl_1, rl_2.....rl_{1674}]$ ▷ all radio link sites
$E_s = \{rl_1 : [d_1..], .., rl_{1674} : [d_1..]\}$ ▷ set of WS-RL distances
$A \leftarrow [\ ]$ ▷ Initialize list to store shortest distance
for *each* rl in RL_s **do**
    $A.append(min(E_s[rl]))$ ▷ add closest rl distance to list
end for
$distancethreshold \leftarrow max(A)$ ▷ maximum threshold distance

with missing features can cause information loss. We therefore adopt 3 strategies in dealing with missing information.

- For records which have missing data for all features for either or both previous days (weather features), we delete those records from the dataset.
- For missing numerical features, filtering on each unique *mlid*, we replace null values with mean value.
- For categorical features, for each *mlid*, we replace missing values with the most frequent category.

Data Encoding and Normalization. As mentioned, our dataset has categorical features, and we cannot use such features in graph neural network training. The common encoding methods are one-hot encoding or binary encoding. One-hot encoding creates an expanded representation with one binary feature per category, whereas binary encoding results in a more compact representation by encoding categories as binary digits. Binary encoding however has a tendency of introducing ordering to the encoded data where there is no ordinality. The categorical features in our dataset have no ordering to them so we used one-hot encoding. The features transformed using one-hot encoding are *type, tip, card_type, adaptive modulation* and *modulation*. We also converted the *True* and

False values for the radio link failure to Boolean format of 1 and 0 respectively. Finally, we normalized our numerical features using standard scaling to improve the convergence of the machine learning model. The scaling was done per each numerical feature and not globally. Feature based scaling preserves the feature independence and interpretability compared to scaling globally.

Handling Data Imbalance. Our dataset is highly imbalanced, with a class ratio of approximately 1:222. To address this, several techniques can be used, such as under-sampling the majority class, over-sampling the minority class, class weight redistribution, and synthetic oversampling [87]. Since under-sampling may lead to loss of valuable information, we opted for synthetic oversampling and class weight redistribution. We applied the SMOTE (Synthetic Minority Oversampling Technique) method to generate synthetic samples for dates with missing records per *mlid* basis, focusing on the minority class. SMOTE generates synthetic samples by interpolating between minority class instances and their k-nearest neighbors. This improved the class ratio to 1:12. During model training, we applied a 1:3 class weight distribution, giving three times more importance to predicting the minority class (failures) compared to the majority class, ensuring better balance in the model's predictions.

Feature Engineering. One-hot encoding increased the dataset's dimensionality and introduced potential redundant features, which slowed model training and could affect accuracy. To address this, we applied feature importance techniques to reduce the feature size and focus on the most impactful predictors. We used correlation-based methods (Pearson and Spearman) and wrapper-based methods to identify the most relevant features. By aggregating the correlation and importance values from these methods, we pruned the feature space down to 50 key features, improving the efficiency and accuracy of the model.

4.4 Proposed Method

4.4.1 Models, Training and Validation

This section describes (i) the proposed GAT model for radio-link failure prediction, (ii) the daily graph construction procedure, and (iii) the training protocol used to address extreme class imbalance.

4.4.1.1 Graph Attention Neural Networks (GAT)

Machine learning models have shown strong promise in prediction tasks in computer networks. GNNs enhance traditional methods by leveraging graph structure to facilitate information transfer between adjacent nodes. In this work, we use a GAT for link failure prediction. GAT enables node-level representation learning using attention mechanisms [88], where node features are updated by aggregating features from neighboring nodes weighted by learned attention coefficients.

Notation: Let $G = (V, E)$ be the graph, where each node $i \in V$ represents a radio link (RL) site with feature vector $\mathbf{h}_i \in \mathbb{R}^F$ and neighborhood $\mathcal{N}_i$. We use H attention heads indexed by h in the first layer (we set $H = 8$). The symbol k is reserved for the k-nearest weather-station construction used in edge creation (we set $k = 3$).

Our model comprises two attention layers followed by a classification block. The first GAT layer applies multi-head attention. For each head h, the node update is:

$$\mathbf{Z}_i^{(h)} = \mathrm{ELU}\left(\sum_{j\in\mathcal{N}_i} \alpha_{ij}^{(h)} \mathbf{W}^{(h)} \mathbf{h}_j\right), \tag{4.1}$$

where $\mathbf{h}_j$ is the feature vector of neighboring node j, $\mathbf{W}^{(h)}$ is a learnable weight matrix,

and $\alpha_{ij}^{(h)}$ is the attention coefficient computed as:

$$\alpha_{ij}^{(h)} = \frac{\exp\left(\text{LeakyReLU}\left((\mathbf{a}^{(h)})^T \left[\mathbf{W}^{(h)}\mathbf{h}_i \| \mathbf{W}^{(h)}\mathbf{h}_j\right]\right)\right)}{\sum_{j' \in \mathcal{N}_i} \exp\left(\text{LeakyReLU}\left((\mathbf{a}^{(h)})^T \left[\mathbf{W}^{(h)}\mathbf{h}_i \| \mathbf{W}^{(h)}\mathbf{h}_{j'}\right]\right)\right)}. \tag{4.2}$$

The output of the first layer is the concatenation of the outputs from all heads:

$$\mathbf{H}_i = \Big\|_{h=1}^{H} \mathbf{Z}_i^{(h)}, \tag{4.3}$$

where $\|$ denotes concatenation. The second GAT layer processes $\mathbf{H}_i$ using single-head attention (same form as above, without head concatenation).

Finally, the output layer applies a log-softmax to produce class log-probabilities:

$$\mathbf{o}_i = \log\left(\text{softmax}(\mathbf{Y}_i)\right), \tag{4.4}$$

where $\mathbf{Y}_i$ denotes the logits from the second GAT layer for node i. We use dropout of 0.6 in both GAT layers.

We use two GAT layers as a practical trade-off between expressiveness and stability on sparse weather-footprint graphs: the first captures local interactions and the second propagates information beyond immediate neighbors without the over-smoothing risk of deeper stacks. We set $H = 8$ attention heads to increase model capacity and training robustness by attending to multiple interaction patterns. The 50-dimensional node feature vector is formed by concatenating RL-KPI/site attributes with short-horizon historical weather context, balancing predictive signal with manageable dimensionality and reduced missing-data amplification versus longer histories.

4.4.1.2 Failure Prediction Graph Construction

After preprocessing, we obtain data samples for 343 unique dates, with measurements associated to 390 weather stations and RL sites (*mlid* pairs) per date. We generate one graph per date; each graph is used as an input instance to the GAT model.

Nodes and node features. Each node represents an RL site. Node features are a concatenation of radio KPIs (e.g., *bbe*, *unavail_second*) and weather attributes (e.g., temperature, wind direction), resulting in a 50-dimensional feature vector per node.

Edges. We associate each RL site with its k nearest meteorological (weather) stations based on geographic distance; in all experiments we use $k = 3$. We then create an undirected edge between two RL nodes if they share the *same set* of k-nearest weather stations.

Let $\mathcal{R}$ denote the set of RL sites and $\mathcal{W}$ the set of weather stations. For each RL site $r \in \mathcal{R}$, we define its associated station set as:

$$\mathcal{S}_k(r) = \mathrm{KNN}_k(r; \mathcal{W}), \tag{4.5}$$

i.e., the set of the k nearest stations to r by geographic distance restricted to stations within the WS–RL association radius d_{th}. We then construct an undirected graph $G = (\mathcal{R}, \mathcal{E})$ where an edge between two RL sites r_i and r_j exists if their station sets match:

$$(r_i, r_j) \in \mathcal{E} \iff \mathcal{S}_k(r_i) = \mathcal{S}_k(r_j), \quad i \neq j. \tag{4.6}$$

Intuitively, weather stations provide discrete samples of a continuous weather field. If two RL sites share the same set of k nearest stations, they fall within the same local measurement neighborhood and therefore are expected to observe highly similar weather conditions (e.g., temperature, wind, precipitation). For example, with $k = 3$,

suppose site A's nearest stations are $\{S_1, S_2, S_3\}$ and site B's nearest stations are also $\{S_1, S_2, S_3\}$; then we add an edge (A, B) because both sites are influenced by the same measured weather footprint. If a third site C has nearest stations $\{S_2, S_3, S_4\}$, it is not connected to A (or B) under this rule because it lies in a different footprint (even though there is partial overlap). This yields a sparse graph that emphasizes strong, localized weather-driven correlation while avoiding long-range or weakly related connections.

Model output. The model predicts a binary label per node: 1 for a radio link failure (positive class) and 0 for no failure (negative class).

4.4.1.3 Training Objective and Optimization

We train the GAT model for 300 epochs using a class-weighted CrossEntropy loss to address the extreme class imbalance. We assign a weight ratio of 3:1 (positive:negative) to control the relative importance of the classes during training. Adam optimizer [89] is used with learning rate $\eta = 0.005$.

4.5 Model Evaluation Results

This section presents the evaluation results of the proposed GAT-based model and traditional models. The dataset was split into training, validation, and testing sets based on time-series and non-time-series data. Data from 2019 was used for training and validation, while 2020 data was reserved for testing to simulate real-world future failure predictions. This ensured that the test data was unseen during training, providing an unbiased evaluation.

The 2019 data was split into 80% for training and 20% for validation, maintaining the class ratio of failure vs. non-failure cases in both sets. The entire 2020 data was used for testing. To address the class imbalance (with only 0.061% failure cases), we applied

SMOTE to oversample the minority class in the training set. This was done solely on the training set to avoid data leakage. Additionally, class-weighted loss functions were used during evaluation to further manage the imbalance.

4.5.1 Benchmark Methods

To evaluate the performance of our proposed GNN-based model, we compared it with widely-used machine learning models: LR, SVM, and LSTM. These models serve as baselines due to their prior application in radio link failure prediction.

Logistic Regression (LR): A simple linear model used for binary classification tasks, where the goal is to estimate the probability of a binary outcome based on input features. During classification, the model calculates a probability for the input and determines the class based on a set threshold which is usually 0.5 [90]. It serves as a baseline for comparison due to its simplicity and efficiency.

Support Vector Machines (SVM): SVM is a supervised learning algorithm that attempts to find the optimal hyperplane separating the classes (failure vs. non-failure) with the maximum margin [91]. In this study, we use SVM with non-linear kernels to model complex decision boundaries.

Long Short-Term Memory (LSTM): LSTM is a type of recurrent neural network (RNN) that is effective for sequence prediction tasks. It was included because of its ability to model temporal dependencies, which are important when predicting future failures based on historical data. **LSTM baseline configuration.** For a fair temporal baseline, we train an LSTM on per-link sequences: for each RL site we form a length-2 sequence from the same engineered KPI/site/weather feature vector used by the GAT (two days of history as two timesteps), and predict the next-day binary failure label. The LSTM output at the final timestep is passed through a fully connected layer to produce

class logits, and the model is trained with the same train/test splits and class-imbalance handling protocol used for the other baselines.

We compare against LR and SVM because they are widely used and strong classical baselines for binary classification on tabular KPI features, are computationally efficient, and provide interpretable reference points. We also include an LSTM baseline to represent sequence modeling and to test whether temporal history alone (without explicit relational structure) can capture weather-driven degradation patterns in KPIs. Together, LR/SVM/LSTM span common operational baselines (classical and temporal deep learning), enabling a controlled evaluation of the incremental value of explicit graph learning. Recent state-of-the-art (SoTA) RLF predictors often introduce additional components (e.g., hybrid GNN–Transformer pipelines, multi-day forecasting modules, or specialized feature-selection/denoising stages) and are typically validated under different datasets, feature sets, and prediction horizons. A fair comparison would require re-implementing these methods and re-tuning hyperparameters on our operator dataset under the same preprocessing, splits, and label definitions; however, detailed training recipes and reference implementations are not consistently available, and architectural differences can confound attribution. Therefore, we adopt representative and reproducible baselines to isolate the contribution of explicit graph structure.

All models are evaluated on the same prediction task: binary failure prediction per RL site per day. We use the same preprocessed feature set (radio KPIs, weather attributes, and clutter class) and the same train/validation/test partitions across models. The proposed GAT consumes the daily graph and produces node-level predictions, whereas LR and SVM use the same node feature vectors without graph message passing. For LSTM, node features are formatted as a short temporal sequence using the same look-back window described in our model. We used 5-fold cross-validation on all models and

calculated the metrics to obtain more robust and reliable results. The cross-validation was done on the train/validation data only to finalize hyperparameters for the trained model.

4.5.2 Evaluation Metrics

We report Precision, Recall, F1-score, Accuracy, ROC–AUC, and PR–AUC for binary failure prediction. Because failures are extremely rare in our dataset, accuracy can be misleading; therefore, we emphasize F1-score and PR–AUC, which better reflect minority-class detection performance under severe class imbalance. ROC–AUC and PR–AUC are computed from prediction scores across all thresholds.

Table 4.3: Performance metrics used for binary failure prediction.

Metric	**Interpretation**
Precision	$TP/(TP+FP)$; reliability of predicted failures
Recall	$TP/(TP+FN)$; fraction of failures successfully detected
F1-score	Harmonic mean of Precision and Recall
Accuracy	$(TP+TN)/(TP+TN+FP+FN)$; can be inflated by class imbalance
ROC–AUC	Area under ROC curve; threshold-free separability measure
PR–AUC	Area under Precision–Recall curve; preferred for rare positives

We also used the ROC curve to visualize the performance of the proposed model. The area under curve (AUC) of ROC curve is the probability score that a randomly sampled positive example has a higher probability than a negative one [92].

4.5.3 Results

4.5.3.1 Baseline comparison

The performance evaluation of the models in Table 4.4 shows that our GNN model outperforms others across key metrics, achieving the highest precision score of 0.642, followed by LSTM (57.2%), SVM (56.8%), and LR (41.7%). While the LR model has the lowest precision and recall, our model exhibits the highest recall value of 81.2%. As noted in Section V.B, accuracy is inflated under severe class imbalance; therefore, we focus on Precision, Recall, F1-score, and AUC metrics to assess minority-class failure detection.

The LR model performs poorly on minority class data. LR achieves a much lower F1-score (39.2%) compared to our GNN model (71.7%), indicating that it struggles to detect failure cases. LR is a linear model, meaning it cannot effectively capture complex spatial dependencies in radio link data. Unlike GNNs, which incorporate historical weather conditions and spatial relationships between radio sites, LR treats each instance independently, missing crucial context. These factors show accuracy alone is misleading in this imbalanced classification problem, and alternative metrics such as precision, recall, and the F1-score better reflect the effectiveness of the model. Our model achieves the best F1-score of 71.7%, surpassing LSTM (60.2%), SVM (58.4%) and LR (39.2%). The F1 improvement of GAT over LSTM is driven primarily by better minority-class retrieval. Compared to LSTM, GAT increases recall from 0.610 to 0.812 (+0.202), meaning it detects substantially more true failure events, while also improving precision from 0.572 to 0.642 (+0.070), indicating fewer false alarms per detected failure. This is consistent with the modeling difference between the two approaches: LSTM learns temporal patterns from each link largely in isolation, whereas GAT performs message passing across sites connected by shared k-nearest weather-station footprints, enabling the model to exploit

cross-site correlations induced by localized weather conditions. In practice, this relational aggregation helps separate subtle failure precursors from normal KPI variability, improving recall without collapsing precision, which explains the higher F1-score.

Table 4.4: Comparison of model performances

Model	Precision	Recall	F1-score	Accuracy	AUC
GAT (our model)	**0.642**	**0.812**	**0.717**	0.971	**0.969**
LR	0.417	0.562	0.392	**0.992**	0.956
SVM	0.568	0.606	0.584	0.960	0.968
LSTM	0.572	0.610	0.602	0.960	0.962

Table 4.5: Cross-validation performance (mean ± std across folds).

Model	Precision	Recall	F1-score	Accuracy	ROC–AUC
GAT (our model)	**0.642±0.021**	**0.812±0.042**	**0.717±0.029**	0.971±0.010	**0.969±0.011**
LR	0.417±0.065	0.562±0.034	0.392±0.065	**0.992±0.013**	0.956±0.061
SVM	0.568±0.047	0.606±0.057	0.584±0.065	0.960±0.024	0.968±0.016
LSTM	0.572±0.067	0.610±0.040	0.602±0.020	0.960±0.021	0.962±0.019

4.5.3.2 Threshold analysis and PR/ROC behavior

We further analyzed our model to explore potential improvements by performing a ROC curve analysis, as shown in Figure 4.2. When the false positive rates are low, our model effectively increases true positive rates while maintaining low false positives, demonstrating satisfactory performance.

In binary classification, labels are assigned based on predicted probabilities, with a typical threshold of 0.5. The results in Table 4.4 are based on this default threshold. However, the threshold can be adjusted to observe the trade-off between precision and recall. Depending on the most relevant performance metric for the task, adjusting the

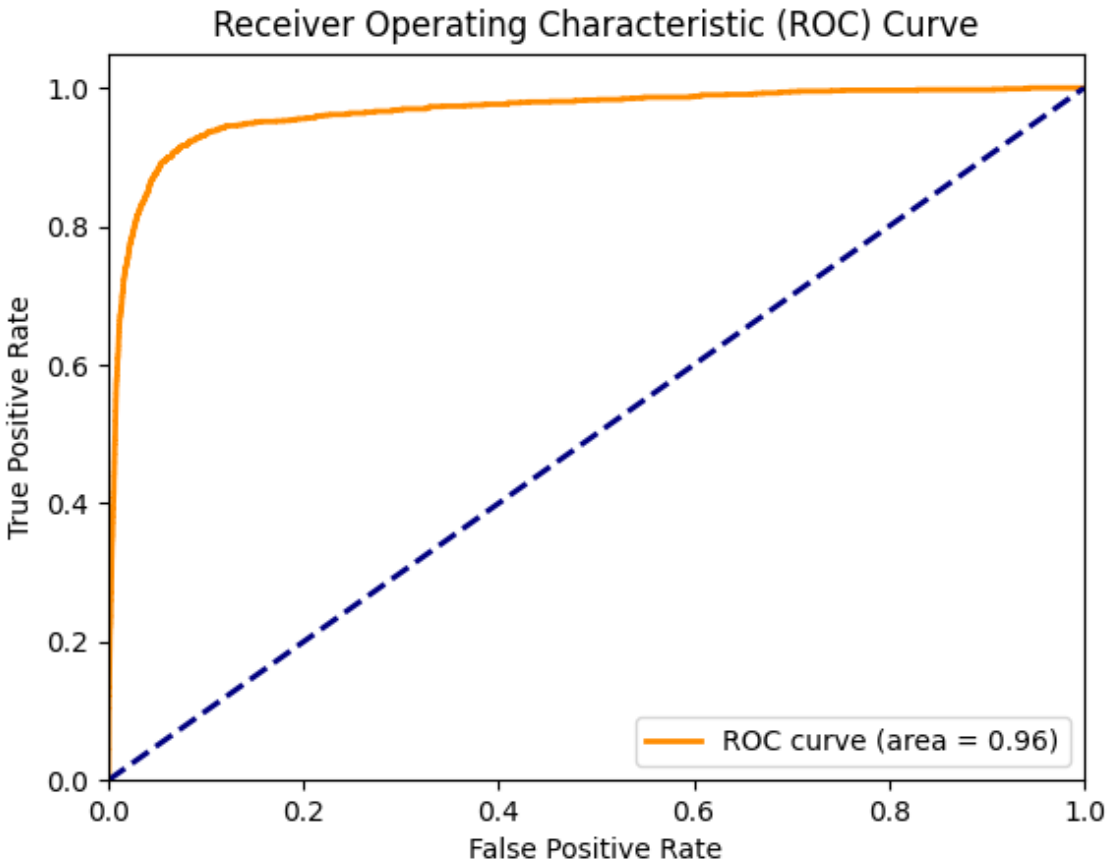


Figure 4.2: ROC curve of our model

threshold allows for better optimization of the model's performance. Figure 4.3 illustrates the impact of varying the classification threshold on our model, providing insight into how different thresholds affect the precision-recall balance.

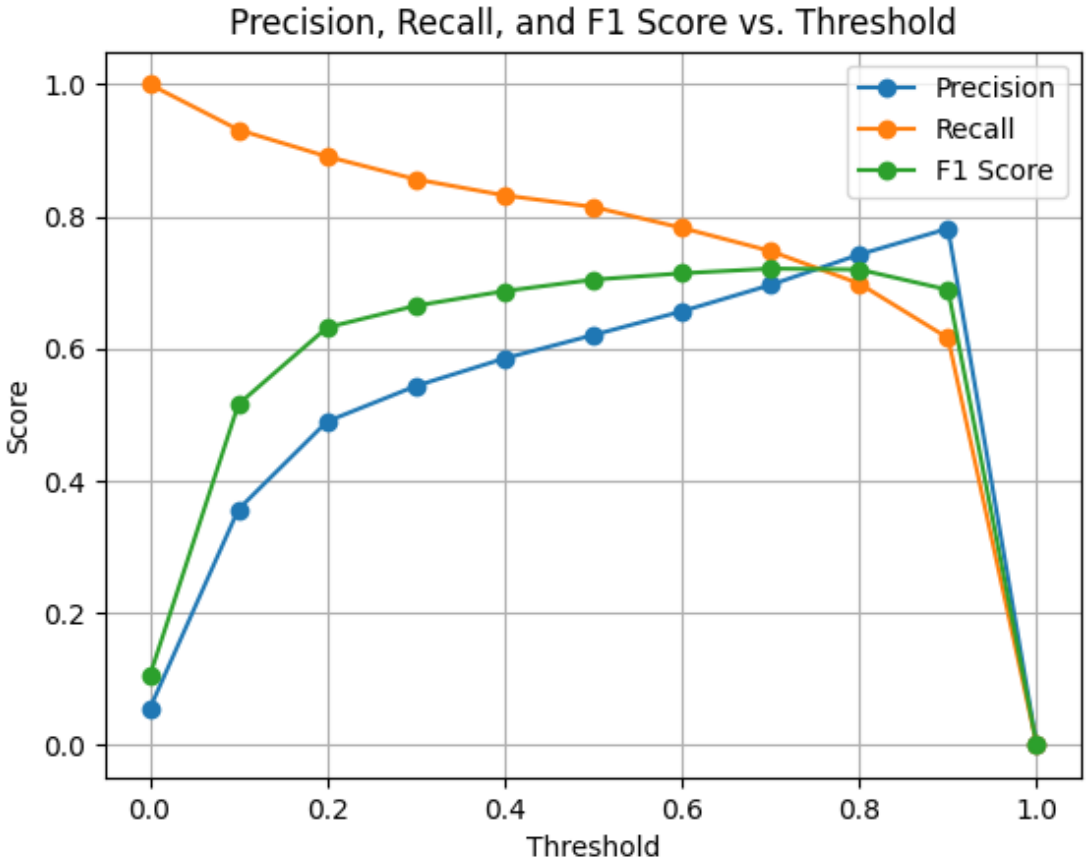


Figure 4.3: Model performance with different threshold values

From the graph, the model starts with a high recall value and then decreases as precision increases. Because missed failures are costly, we prioritize recall while selecting a threshold that keeps the alert volume (false positives) within an operator-acceptable budget. In our data, a threshold of 0.76 yields a more balanced operating point (Precision 73.2%, Recall 73.6%, F1 73.4%).

4.5.3.3 Discussion and Key Takeaways

The effectiveness of GNNs, particularly GAT, over traditional machine learning models stems from their ability to capture spatial dependencies, handle graph-structured data, and leverage adaptive feature aggregation through attention mechanisms.

- **Capturing Spatial Dependencies**: Traditional machine learning models treat input data points as independent observations, ignoring spatial correlations. However, radio link failures are often influenced by neighboring radio sites and their environmental conditions. GNNs propagate information across nodes in the network, enabling a more informed failure prediction.

- **Adaptive Feature Aggregation**: Unlike traditional models, which apply uniform transformations to all input features, GAT employs attention mechanisms that dynamically weigh neighboring nodes based on their importance. This results in a more targeted and effective feature extraction process.

- **Robustness to Data Imbalance**: The extreme class imbalance in radio link failure data can lead traditional models to favor the majority class, reducing sensitivity to rare failure instances. GNNs leverage neighborhood aggregation, smoothing feature representations and improving classification performance on minority classes.

- **Temporal and Spatial Awareness**: While LSTM models capture temporal dependencies, they lack explicit spatial modeling. Our GNN-based approach effectively combines both spatial (neighboring radio links) and temporal (historical weather data) relationships, making it a more holistic prediction framework.

To ensure a fair comparison, we evaluate the computational cost of the proposed GNN-based model and baseline methods in terms of training time, inference time. Table

4.6 presents the results obtained by running each model on an NVIDIA RTX 3090 GPU with 24GB VRAM and an Intel Xeon CPU.

Table 4.6: Comparison of model computational cost

Model	Training Time (s)	Inference Time per Sample(ms)
GAT(our model)	∼1800	2.5
LR	∼8	0.1
SVM	∼50	0.5
LSTM	∼910	1.2

From Table 4.6, LR is the fastest model due to its low complexity and linear nature, which makes it computationally efficient but unsuitable for complex, spatially structured problems. SVM takes longer to train and infer due to its dependence on kernel functions. LSTM has a significantly higher training cost because it processes sequential data, but it does not leverage spatial dependencies like GNNs. Our GNN-based model (GAT) has the highest computational cost due to graph convolutions and attention mechanisms, but achieves the best predictive performance for the minority class. Despite its higher computational cost, the GNN model's ability to capture spatial and temporal relationships makes it the most effective choice for real-world deployment, where prediction accuracy is critical.

4.5.4 Ablation Study: Impact of Removing Class Balancing Methods

4.5.4.1 Objective and Setup

In this ablation study, we retrained the GAT model *without any class-balancing techniques* (no SMOTE oversampling and no class-weighted loss), while keeping all other

components identical (features, graph structure, architecture, optimizer, and train/test splits). This exposes the model to the true failure rate of ∼0.061% without compensation, resulting in an extremely imbalanced training scenario. Table 4.7 compares the performance of the GAT *with* versus *without* class balancing.

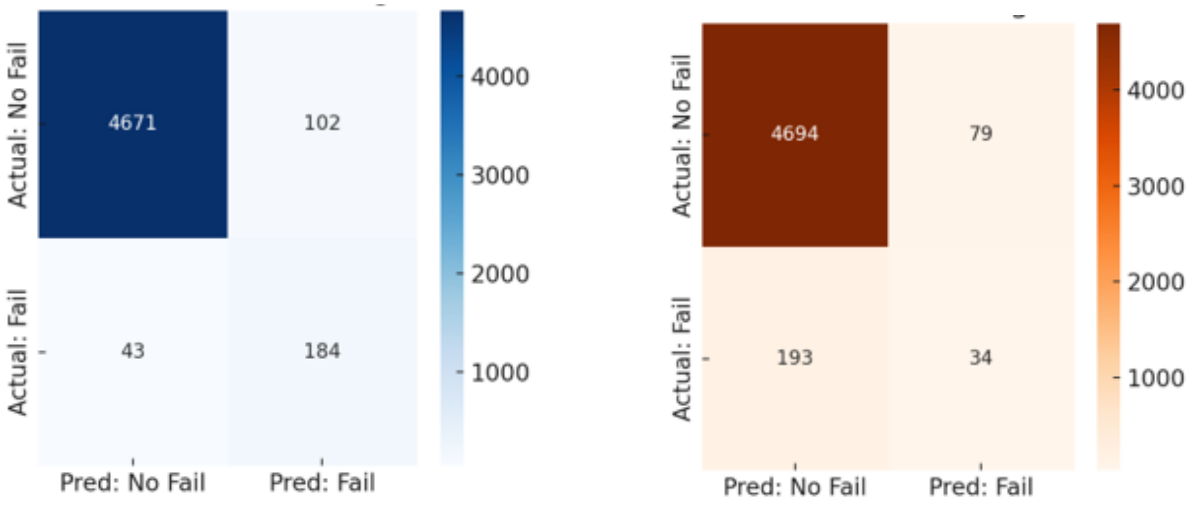
Table 4.7: GAT Performance with vs. without class balancing

Model	Precision	Recall	F1-score	Accuracy	ROC-AUC	PR-AUC
With Balancing	0.642	0.812	0.717	0.971	0.969	0.62
No balancing	0.30	0.15	0.21	0.995	0.93	0.11

4.5.4.2 Quantitative Results

As shown in Table 4.7, removing class balancing causes a drastic drop in predictive performance. The unbalanced GAT captures only ∼15% of the failure instances (Recall ≈ 0.15), a sharp decline from 81.2% with balancing. Precision also drops from 64.2% to around 30%, indicating that when the unbalanced model predicts a failure, the prediction is incorrect over two-thirds of the time. Consequently, the F1-score collapses to ∼0.20 without balancing, compared to 0.717 with balancing. This contrast indicates that training on raw imbalanced data makes the model nearly blind to the minority class, detecting only a small fraction of failures while misclassifying many of the few predicted ones.

Figures 4.4a & 4.4b further illustrate the difference in model behavior. The balanced model identifies far more failures (184 true positives vs. only 34 without balancing), while the unbalanced model predicts "no failure" for almost everything, missing 85% of actual failures. In the unbalanced case, the few failure alarms (34) come at the cost of 79 false positives, whereas the balanced GAT catches 184 failures with 102 false alarms. The



(a) With Class Balancing (b) Without Class Balancing

Figure 4.4: Confusion matrices for the GAT with and without class balancing.

color intensity highlights that the unbalanced model's predictions are overwhelmingly biased toward the majority class, with almost all predictions concentrated in the "No Fail" column.

As discussed in Section V.B, the high accuracy in the no-balancing setting is dominated by true negatives and does not reflect failure-detection utility; therefore we interpret the degradation primarily through Precision/Recall/F1 and PR–AUC. The ROC–AUC and PR–AUC further illustrate these shortcomings. Without balancing, the ROC–AUC drops from ∼0.97 to roughly 0.90 (Table 4.7), indicating reduced discrimination between failures and non-failures. More importantly, PR–AUC plunges to ∼0.10 without balancing, an order-of-magnitude lower than the ∼0.6 achieved with balancing. Given the extreme class skew, PR–AUC is particularly informative: a near-zero PR–AUC indicates that across threshold settings, precision remains very low for any achievable recall. In practical terms, without balancing, the GAT provides almost no useful early-warning signal—either it triggers almost no alarms (yielding high precision but near-zero recall), or increasing sensitivity floods the system with false alarms.

4.5.4.3 Operational Interpretation and Takeaways

From a model behavior perspective, training on unmodified imbalanced data also affects the GAT's attention mechanism and message-passing dynamics. In a GAT, each node aggregates information from neighbors using learned attention weights. When failure examples are extremely rare, training is dominated by majority (non-failure) patterns, and the model learns to minimize loss by prioritizing those abundant normal patterns. Without class-weighting pressure to counteract imbalance, the attention mechanism is likely to underemphasize minority-class signals, and message passing can dilute rare failure precursors through aggregation with many more operational neighbors. Only with balancing does the model learn to assign higher importance to informative minority patterns.

These findings highlight why class-balancing methods are critical for real-world deployment of the RLF prediction model. In a live network, failures are infrequent, yet missing a looming failure can be extremely costly (e.g., unexpected downtime and financial losses). A model that detects virtually no failures, as the unbalanced GAT does, would be ineffective for proactive maintenance. Conversely, excessive false alarms can overwhelm operators and erode trust. Achieving a workable balance between recall and precision requires the model to learn minority patterns meaningfully. In our case, SMOTE oversampling and class-weighted loss were essential to give failure instances sufficient influence during training. With balancing, the model captured over 80% of failures with precision above 60%, enabling proactive failure prediction; without these measures, the model would provide a false sense of security with $\sim$99% accuracy while most actual failures slip through undetected.

In summary, this ablation study demonstrates that removing class balancing devastates the GAT's ability to predict rare radio link failures. Precision and recall collapse to

untenably low levels without oversampling and class-weighting, even as accuracy appears high due to skewed class distribution. These outcomes reinforce that class balancing is not merely a training detail but a necessary component for deploying GNN-based solutions in highly imbalanced domains.

4.5.5 Ablation Study: Imbalance handling methods

To rigorously evaluate our class-imbalance strategy, we conducted an ablation study comparing multiple techniques for handling the extreme imbalance (0.061% positive class) in model training. All experiments used the same GAT architecture, features, optimizer, 2-day historical window, and 2019/2020 train-test split as the baseline. The only change in each case was the method to address class imbalance. We evaluated:

- using only a class-weighted loss (no resampling)
- using focal loss (γ=2, with tuned α for class balance)
- random undersampling of the negative class
- oversampling via ADASYN
- SMOTE with Tomek links
- SMOTE with Edited Nearest Neighbors (ENN), and
- our baseline of SMOTE + class-weighted loss.

Table 4.8 summarizes key metrics:precision, recall, F1-score, accuracy, ROC-AUC and PR-AUC for each method.

Table 4.8: Comparison of Class Imbalance handling methods

Method	Precision	Recall	F1-score	Accuracy	ROC-AUC	PR-AUC
Class-Weighted Loss only	0.55	0.60	0.57	0.990	0.950	0.55
focal loss (γ=2, tuned α)	0.56	0.80	0.66	0.970	0.960	0.62
Random Undersampling	0.30	0.80	0.44	0.950	0.940	0.40
ADASYN Oversampling	0.62	0.80	0.70	0.970	0.968	0.68
SMOTE + Tomek Links	0.67	0.80	0.73	0.976	0.972	0.75
SMOTE + ENN	0.70	0.78	0.74	0.978	0.970	0.76
Our baseline	0.64	0.81	0.72	0.971	0.969	0.72

4.5.5.1 Class-Weighted Loss Only

Using only a weighted loss with no oversampling did improve failure recall modestly over an unweighted model, but it still missed nearly half of failure instances (recall approx. 60%). The weighted loss approach achieved the highest overall accuracy (99%) by largely predicting the majority class, but this accuracy is misleading. High accuracy stems from trivially classifying most links as non-failures, which does little to identify actual failures. Precision for the minority class was moderate (55%), indicating that when failures were predicted, over 40% were false alarms. Overall, the F1-score remained relatively low (57%), demonstrating that weighting alone was insufficient. This is expected, as without oversampling the model sees extremely few positive examples, and even a cost-sensitive loss cannot fully compensate. Prior work also notes that purely weighting the loss can still be dominated by the abundant class. The strength of this approach is its simplicity and avoidance of any synthetic data; it does not risk introducing noise or bias from generated samples. However, the downside is underwhelming recall, the model is still biased toward predicting "no failure" for most instances to minimize loss. In our context, missing over 40% of failures is unacceptable, so this method alone was not adequate.

4.5.5.2 Focal loss

Applying focal loss (γ=2) yielded a significant boost in failure detection compared to simple weighting. By down-weighting easy majority samples and focusing training on hard, misclassified examples, the focal loss drove the model to catch far more failures (recall approx. 80%). In fact, focal loss achieved nearly the same recall as our oversampling baseline, indicating it effectively countered the class imbalance. Precision, however, was only 56%, meaning almost half of predicted failures were false positives. This reflects the focal loss trade-off: it aggressively learns minority class features increasing true positives at the cost of more false alarms. The F1-score (66%) improved over the basic weighted loss, but still fell short of the SMOTE-based methods. We also found that tuning the α parameter, class weight in focal loss was important: a moderate α gave the best balance. Too high an emphasis on the minority (large α) made the model output many false failure alerts, whereas too low α undercut recall. Overall, focal loss proved useful in emphasizing minority instances and is a viable alternative to oversampling. Its strength is that it avoids generating synthetic data while achieving high recall. However, it required careful hyper-parameter tuning, and even then, its precision-recall balance (62%) was slightly below the SMOTE approaches. In summary, focal loss can substantially improve sensitivity to rare failures, but in our experiments it did not surpass the combination of SMOTE with class weighting.

4.5.5.3 Random undersampling

As expected, randomly undersampling the majority class degraded overall performance. By discarding a large portion of "no failure" examples, this method balanced the class ratio in training but at the cost of losing significant information. The model's failure recall was indeed high (80%), since the training data contained a much higher fraction of

failures. However, this came with an explosion in false positives: precision plummeted to approx. 30%. Consequently, PR-AUC was the lowest of all methods (40%), and F1-score sunk to approx 44% despite the high recall. Notably, overall accuracy dropped to 95%, the lowest among all methods, because the model frequently cried wolf on the majority class. These results align with the known drawbacks of undersampling: removing majority samples can eliminate important counter-examples and make the classifier overly optimistic about the minority class [72]. In effect, the GAT model trained on undersampled data lost generalization on normal conditions, yielding poor precision. Although undersampling is simple and avoids synthetic data, its severe information loss made it unsuitable for our problem, especially where false alarms are costly. We confirm that undersampling is not effective under extreme imbalance, consistent with prior findings [72].

4.5.5.4 ADASYN oversampling

Using ADASYN (Adaptive Synthetic Sampling) produced results very similar to plain SMOTE. ADASYN generates synthetic minority examples preferentially in harder-to-learn regions where minority data is sparse or borderline. In our experiments, ADASYN achieved roughly the same failure recall (80%) as SMOTE, with a minor difference in precision. We observed a slight trade-off: ADASYN yielded a marginal increase in recall by covering more minority variants but also introduced a bit more noise, lowering precision to 62%. The net effect was an F1-score (70%) essentially on par with SMOTE. In terms of AUC metrics, ADASYN matched SMOTE's ROC-AUC (97%) and had a nearly identical PR-AUC (68–70%). This indicates that ADASYN's targeted oversampling did not significantly outperform basic SMOTE for our data. A possible reason is that our minority class, though extremely small, may not have had highly complex sub-regions

for ADASYN to exploit, the simpler SMOTE interpolation was sufficient to cover the space. Additionally, any synthetic oversampling risks adding some noisy samples, and ADASYN is no exception. Its strength lies in focusing on more challenging minority samples potentially improving generalization, and it did help maintain high recall. However, we did not see a clear advantage over SMOTE; both approaches substantially boosted the model's ability to detect failures. We conclude that ADASYN is a viable alternative to SMOTE, achieving comparable precision-recall performance but it did not materially change the outcome in our case.

4.5.5.5 SMOTE + Tomek Links

Combining SMOTE oversampling with Tomek links yielded notable improvements in precision. Tomek link removal identified instances where a minority and majority sample are very close, potentially overlapping and removed the majority instance. By cleaning these borderline cases after oversampling, the model was less confused by ambiguous "no-failure" examples. This is reflected in precision rising to 67%, the second-highest among all methods. Importantly, this gain came without sacrificing recall, the recall remained about 80%, on par with baseline SMOTE. Consequently, the F1-score improved to 73%, and the PR-AUC (75%) was the highest of all tested methods slightly edging out the baseline. The removal of Tomek link pairs effectively reduced class overlap and noise, leading to a more separable boundary and fewer false positives. We also note a small increase in accuracy (97.6%) relative to baseline, thanks to those false positives being eliminated. The strength of SMOTE+Tomek is clearly in refining the oversampled dataset: it keeps synthetic minority examples but purges confusing majority examples, yielding a cleaner training signal. A potential weakness is the added complexity, an extra data cleaning step – and the possibility that if a genuine failure case was very border-

line, its nearest neighbor being a majority could cause it to be removed. In our results, SMOTE+Tomek provided the best precision-recall balance overall. This method shows that a hybrid approach can outperform oversampling alone, corroborating recommendations in imbalanced learning literature [93].

4.5.5.6 SMOTE + ENN

This hybrid method combines SMOTE with Edited Nearest Neighbors (ENN) cleaning, and it achieved the highest precision (70%) of all methods. After oversampling, ENN removes any sample (majority or minority) that is misclassified by its k-nearest neighbors [93], thereby excising noisy points and regions of overlap. The result was a very "clean" training set, which made the model's predictions more conservative and precise. We see this in the metrics: false positives were greatly reduced, pushing precision to 70% and accuracy to 97.8%. The trade-off was a slight drop in recall (to 78%). It appears that ENN may have removed a few minority instances or synthetic samples that were in majority-dominated neighborhoods as noise. Those could have included some useful hard examples, so the model missed a few failures that the baseline would catch. Despite this, the F1-score (74%) and PR-AUC (76%) were the highest of all methods, marginally above SMOTE+Tomek. In essence, SMOTE+ENN delivered a very robust classifier with fewer false alarms, at the expense of a small amount of sensitivity. This trade-off might actually be desirable in some deployments, but in our failure prediction context we prioritized recall. It is worth noting that SMOTE+ENN, like SMOTE+Tomek, adds more complexity to the pipeline and requires careful implementation to ensure no data leakage from cleaning on test data. Nonetheless, our ablation confirms the literature's claim that SMOTE+ENN produces a balanced and clean dataset for training [93], leading to strong performance. The choice between Tomek vs. ENN may come down to whether

one prefers slightly higher recall (Tomek) or higher precision (ENN).

4.5.5.7 Baseline (SMOTE + class weights)

Our originally reported method, SMOTE oversampling:increasing minority ratio to 1:12 combined with a 3:1 class-weighted loss proved to be a well-rounded compromise. It achieved an F1-score of 71.7%, higher than all single-technique methods (weighted loss or focal or undersampling) and only marginally below the best hybrid methods. Notably, it maintained the highest recall (81.2%) among all oversampling techniques, while attaining a respectable precision of 64.2%. This balanced precision-recall profile (72%) means the model catches the vast majority of failures without overwhelming operators with false alarms. In practice, this balance is crucial: missing fewer failures (high recall) was our top priority, since undetected failures directly translate to outages and losses [94]. At the same time, a decent precision helps ensure that failure alarms are actionable and not ignored as noise. The baseline method's recall was on par with focal loss and undersampling, but with far fewer false positives than undersampling and a higher F1 than focal loss. Compared to the more complex SMOTE+Tomek/ENN, our baseline had slightly lower precision, but it kept every real failure example in the training data and thus retained the maximal recall. Given that the improvements in F1-score from Tomek or ENN were on the order of only approximately 2%, we consider our simpler SMOTE+weights approach to be justified as the primary choice. It provides a strong baseline that already significantly mitigated class imbalance by oversampling to 1:12 ratio and weighting, and our ablation confirms that no other method produced a dramatically better outcome. In summary, oversampling the minority class with SMOTE combined with a moderate class-weighted loss offered the most effective boost to our GAT's failure prediction capabilities. This strategy improved minority class sensitivity while preserving

model generalization to the majority class, resulting in the most effective GNN-based early failure predictor under extreme imbalance.

4.5.6 Practical Deployment of the Proposed ML System

The proposed GAT based RLF prediction system is designed for seamless integration into a telecom operator's network management workflow [85]. At a high level, the system ingests two categories of data: real-time network KPIs from 5G RAN nodes, e.g. error rates, availability counters and weather measurements from meteorological sources, both current and recent historical values. These inputs are processed and merged into a graph-structured dataset with each node representing a radio link including its site attributes and performance metrics and edges denoting relationships, e.g. links sharing the same nearest weather stations. For each link node, a feature vector is constructed that includes the latest day's KPIs and site information, along with two days of historical weather features aggregated from the k closest weather stations. This 2-day context ensures the model captures short-term temporal trends while avoiding excessive missing data issues. The GAT model then performs inference on this graph, producing a probability of failure for each link which is binary classification of failure or no failure. The prediction outputs are fed into an alerting module that flags high-risk links based on a configurable probability threshold. Finally, results are surfaced to operators via dashboards and can trigger automated mitigation actions. Figure 4.5 illustrates the end-to-end architecture, from data sources through model inference to outputs and feedback loops.

The proposed GNN-based failure prediction model is designed to integrate into an automated network monitoring system that enables real-time prediction of radio link failures. The system can be implemented as follows:

Figure 4.5: End-to-end pipeline for GAT-based RLF prediction: data sources (topology, weather, KPIs/logs) → ingestion/cleaning → feature engineering → graph construction → GAT inference (per-link failure probability) → thresholded alerts for NOC dashboards and automated mitigation, with an MLOps loop for monitoring, feedback, retraining, and CI/CD deployment.

4.5.6.1 Deployment Environment

The RLF prediction system can be deployed in multiple environments depending on operator preferences and constraints. In all scenarios, the core GAT model and data pipeline remain consistent with the original design. We briefly discuss three representative deployment models:

4.5.6.1.1 Centralized NOC Deployment In a centralized deployment, the inference engine is hosted in the operator's Network Operations Center (NOC) or a private data center. All KPI data and weather feeds are streamed or batch-transferred to this central location for processing. This setup aligns with traditional telecom OSS (Operations Support Systems) environments. The model effectively becomes an intelligent extension of the NOC's monitoring platform. The advantage is that data remains on-premises addressing data sovereignty and security concerns and integration with existing NMS (Network Management System) databases and dashboards is direct. The GAT model processes incoming data and generates alerts at the NOC in near real-time. Given the optimized model inference time of approximately 2.5 ms per link prediction on GPU hardware, a central server can handle thousands of links with low latency. While model inference is sub-second, the overall end-to-end latency budget of under 5 minutes, as stated in our SLOs, is dominated by upstream data ingestion, cleaning, and feature aggregation steps, ensuring timely alerts for operators. Centralized deployment simplifies maintenance and monitoring of the ML system but requires sufficient uplink bandwidth from cell sites to NOC for continuous data feeding. It is well-suited for operators who already centralize data collection and want full control over the prediction service within their private infrastructure.

4.5.6.1.2 Cloud-Based Inference(Public/Private Cloud) For operators embracing cloud infrastructure, the RLF prediction engine can be deployed to a public or private cloud environment. In this model, data from the RAN and possibly from third-party weather APIs is securely transmitted to a cloud platform where the GAT inference service runs. A private telco cloud or hybrid cloud may also be used to keep sensitive data on trusted infrastructure. Cloud deployment offers elastic scalability. The provider can automatically scale compute resources to handle spikes in the number of links or higher-frequency inference without on-premises hardware upgrades. For example, the 300+ daily graph instances used in training could be processed in parallel in a cloud environment to achieve high throughput. The model's computational cost is justified given its predictive accuracy on rare failures, and cloud GPU instances can be provisioned to meet these performance needs on demand. A potential challenge is ensuring compliance with data privacy and telecom regulations when exporting network data to an external cloud. Thus, strong encryption and access control are mandatory, and often a VPN or dedicated line is used for data transfer to the cloud. Many operators prefer a "cloud-native" approach using containerized microservices for the data pipeline and model, enabling continuous deployment and easy integration with cloud-based data lakes and analytics tools.

Cloud deployment also facilitates integration with advanced cloud AI services, for example, managed databases for time-series KPI storage or serverless functions for triggering retraining, accelerating development. However, network latency between cell sites and the cloud must be low enough to allow timely predictions. In practice, since our system predicts failures on the order of hours in advance using daily aggregated features, even a few seconds of cloud transit latency is acceptable. This makes cloud inference viable for proactive maintenance use cases, as long as robust security measures are in

place to protect the in-transit and at-rest data.

4.5.6.1.3 Edge or Hybrid Edge-Cloud Deployment In scenarios where ultra-low latency or reduced backhaul traffic is desired, the system can be partially deployed at the network edge on a Multi-access Edge Computing (MEC) server or an O-RAN near-RT RIC controller with a hybrid coordination to the central cloud/NOC. In an edge deployment, preprocessing and inference occur physically near the RAN sources, e.g. a MEC node at a base station hub could run a lightweight instance of the GAT model to predict link failures for the local clutter of sites. This yields very fast detection and allows immediate local remedial actions such as switching frequencies or triggering local redundancy mechanisms without waiting for central analysis. It also conserves bandwidth by avoiding raw data streaming: instead of sending all KPI and weather data to a central server, the edge device computes failure probabilities and only sends alerts or summarized risk scores upstream. Our GAT model is relatively compact and can run on modest hardware, but care must be taken at the edge due to typically more limited compute resources. It may be beneficial to deploy only the inference runtime at the edge with the model pre-trained and periodically updated from the cloud, a strategy supported by the model's small inference footprint.

In practice, an edge-cloud hybrid could work as follows: the central system aggregates global data and periodically retrains or fine-tunes the model incorporating new failure examples, then pushes the updated model to edge nodes. The edge nodes handle inference in real-time and perhaps the first level of alerting for rapid self-healing responses, while also forwarding prediction results to the central NOC/cloud for broader analysis and visualization. This aligns with modern telecom architectures where AI-driven control loops are split between edge (near-real-time control) and central (longer-term analytics). A known concern in edge inference is managing remote software updates and reliability

Table 4.9: Summary of deployment scenarios for RLF prediction.

Scenario	**Latency**	**Control**	**Scalability**	**Cost / trade-offs**
Centralized NOC	Medium–High (backhaul + aggregation)	High (central governance)	Medium (central bottlenecks)	Lower centralized hardware/server cost; higher WAN usage; potential central congestion
Cloud	Medium (internet + service)	Medium (shared platform)	High (elastic compute)	OpEx-based; strong scaling; data residency / egress considerations
Edge	Low (local inference)	High (local autonomy)	Medium (fleet management)	Higher CapEx; best for tight SLOs; requires remote updates/monitoring

of connections [94]. Our system mitigates this by using a containerized model package that can be remotely updated when a new model version is available with fallback to previous model if update fails, and by designing the edge component to be stateless so it can recover quickly from restarts or connectivity losses. The hybrid model thereby combines the responsiveness of edge AI with the global view and heavy compute of cloud AI.

4.5.6.2 Data Ingestion and Graph Construction

4.5.6.2.1 Data Pipeline We ingest two mature operator data streams: RAN KPIs and meteorological feeds on an hourly/daily cadence via the NMS and public station/APIs [3]. A lightweight preprocessing layer aligns timestamps, reconciles units, normalizes features, and imputes short gaps last-observation or linear interpolation, with automated validation to quarantine outliers, counter resets, and schema drift before feature materialization. The result is a clean, continuously refreshed feature store usable for both batch (daily) and streaming (hourly) runs.

4.5.6.2.2 Graph Feature Aggregation For each radio link, we attach local weather context by mapping to its k nearest stations and aggregating statistics for the current

day and the previous two days—an empirically chosen horizon that preserves signal while limiting missingness. Daily inference therefore uses past 24h KPIs plus 2 previous days weather aggregates. Intra-day operation applies the same logic over a sliding ∼48h window. Edges are created between links sharing the same nearest-station set, capturing co-exposure to weather and producing locality clusters. This proximity graph is largely static but can be augmented with dynamic ties such as co-degradation correlations when available. The pipeline emits an adjacency plus node-feature matrix per scoring epoch, enabling low-latency GAT inference. In streaming mode, Kafka or Flink/Spark join KPI and weather events, update cached station mappings, and incrementally refresh node features, while a thin microservice maintains the in-memory graph snapshot for scoring.

4.5.7 Network Operator Dashboard and Explainability

Integrating the prediction outputs into an operator dashboard is key for practical adoption. The dashboard should present a visual overview of the network's health with regards to impending link failures, in a format familiar to NOC engineers. For example, the interface might highlight links on a geographic map which can be color-coded by failure risk, or list the top N at-risk links with their probabilities. The system can be designed to plug into existing network management GUIs. Several NOC tools like Ericsson ENM [95] or Nokia NMS [96] allow custom panels or data import. The Operator Dashboard for our system would display each alert along with contextual information: the link's location and attributes, its recent KPI trends, and the relevant weather conditions. This context helps engineers validate and prioritize alerts. For instance, an alert might show "Link ID 123 in Region Alpha – 85% predicted failure risk. Rainfall last 24h: 50mm (extreme), Link Error Seconds: 120 (above norm)". Providing these details alongside the alert draws a direct line between model inputs and prediction, making it

more explainable and actionable.

4.5.7.1 Explainability

Because GAT models can be seen as black-box by operators, we emphasize explainability features in deployment. One approach is to leverage the GAT's attention weights to indicate which neighboring nodes or which features contributed most to a given prediction. For example, if the model's attention mechanism heavily weighted a particular neighbor link that recently failed, the system can highlight that "this link is considered high-risk partly because a nearby link just experienced a failure under similar weather". Similarly, by analyzing feature importance (e.g. via SHAP values), the dashboard could display a rationale like "High humidity and rapidly rising block error rate are driving this prediction". Such explanations build operator trust and also guide remedial action, e.g. if the cause is weather, one response; if it is purely a KPI trend, another. In critical infrastructure, having a clear reason behind an AI alert is often as important as the alert itself [94]. Therefore, our deployment includes an explainability module that post-processes the model output to produce human-understandable insights. We note that GNN explainability is an active research area, and tools like GNNExplainer or GraphLIME could be integrated to identify which parts of the graph influenced the result. However, even simple measures, such as showing the values of the top features for the link versus normal ranges, can add clarity.

The dashboard also supports feedback input from operators, creating a two-way interaction. If an operator disagrees with or resolves an alert, they can annotate it e.g. "false alarm" or "investigated – issue found in antenna". These annotations are captured for retraining and continuous improvement. The user interface is thus not only a monitoring tool but part of the human-in-the-loop feedback mechanism.

4.5.7.2 Performance, SLOs and Scalability

In a production telecom environment, the RLF prediction system must meet strict Service Level Objectives (SLOs) for latency, accuracy, and availability.

4.5.7.2.1 Latency We retain the original performance envelope: millisecond-level per-link inference on GPU, enabling near–real-time scoring. End-to-end latency targets ¡5 min for streaming updates so KPI/weather shifts can trigger timely alerts; in batch mode, daily runs complete before business hours.

4.5.7.2.2 Predictive quality Live performance should match validation of F1-score of 0.71 and recall of 0.7 as attained by our model. Operators may specify: $\geq$70% of failures predicted in advance with FP rate $\leq$ X. Threshold tuning using PR curves and continuous post-hoc auditing of missed events enforce these targets.

4.5.7.2.3 Throughput and scalability The system is built to scale to large networks (hundreds or thousands of links). GNNs can become computationally heavy as graph size grows, but our approach of splitting by day i.e. one graph per day localizes graph size. In our dataset, each daily graph had on the order of a few hundred nodes. A nationwide network could be partitioned by region or by time to keep graphs tractable. Scalability testing indicates that inference can be parallelized across multiple GPUs or distributed across machines if needed, for example, each region's links could be processed on a separate worker. The cloud deployment scenario particularly eases scaling: additional compute instances can be spun up to handle regional graphs in parallel, and then shut down to save cost. The model's memory footprint (GAT with two layers and attention heads) is modest, and even training was feasible in $\sim$30 minutes on a single GPU based on our model results. For an online system, we anticipate retraining to be needed

perhaps monthly, which is easily scheduled in a scalable environment without impacting the live inference service.

We also consider system capacity in terms of how many predictions per second can be produced. If run as a continuous service, our inference engine can process many links per forward-pass by utilizing batch processing on GPU. The measured 2.5 ms "per sample" inference time shown in Table 4.6 implies that in batches of, say, 100 links, a GPU might handle the batch in a few hundred milliseconds. Thus, even networks with 10,000 links could be scored in well under an hour on one GPU, or faster with parallelization, satisfying daily prediction needs. If we go to per-hour predictions, we may distribute the workload or use a more powerful machine, but it remains within feasible bounds. The system's design allows horizontal scaling by simply adding more parallel model workers and partitioning the input data stream.

4.5.7.2.4 Reliability and Availability The inference service should be highly available targeting 99.9% up-time or higher since it becomes part of the operations tool-chain. We could deploy redundant instances of the model service: active-active or active-standby so that failure of one instance does not drop coverage. Moreover, since this is a prediction system, occasional delays or a single missed run might be tolerable, but we aim to minimize downtime to not miss warning opportunities. Automated monitoring of the pipeline which is the heartbeat of data flow and model responses would be in place to alert if the system is not functioning.

4.5.7.3 Automated Mitigation and Integration

Beyond raising alerts, an advanced deployment can leverage predictions for automated mitigation, effectively using the model in a closed-loop control system to preempt failures. Our design allows the predictive engine to feed into self-healing mechanisms in

the network. For example, if a link is predicted to fail with high probability, the network could proactively reroute traffic from that link to alternate paths if available before the failure actually happens. This could be orchestrated via the Software-Defined Networking (SDN) controller or through the RAN intelligent controller interfaces. Another action could be temporarily increasing transmission power or adjusting antenna tilt for a vulnerable link if heavy rain is predicted, to bolster the link budget. Such policy-driven responses can dramatically reduce impact of failures.

However, automated actions must be done carefully to avoid unnecessary changes. We could implement a policy guardrail as suggested by industry best practices [85]: a predicted failure triggers automation only if certain conditions are met, e.g. the confidence is very high or multiple signals concur. And even then, the actions are reversible and safe. In fact, one can create a conditional plan: if the model says link X has 90% chance of failing in next hour, then prepare a backup route and issue a warning to any connected systems, but only execute the reroute if link X's signal quality actually starts dropping beyond a threshold. This combination of predictive and reactive criteria yields a robust self-healing loop with minimal service disruption.

The system could also be integrated with OSS (Operations Support Systems) and BSS (Business Support Systems) for seamless workflow insertion. OSS integration means the predictions appear in the same console that handles existing alarms and performance data, as earlier described for the dashboard. We could use stable APIs or data exchange formats to ensure our system doesn't break legacy tools [97]. For instance, the model can expose a REST API endpoint which the OSS periodically calls to get current risk levels, or it can dump results into a database table that the OSS GUI reads. On the BSS side, predictive insights can influence customer-facing considerations: for example, if a high-risk failure is predicted on a cell serving a corporate client, the account management

system might proactively inform the client of potential service impact or apply credits as per SLA agreements. Similarly, if certain failures carry regulatory reporting obligations, those processes can be initiated earlier thanks to the prediction. Our system can trigger the creation of trouble tickets/work orders automatically when a threshold is passed, sending a ticket to the field maintenance teams to inspect or replace equipment identified as likely to fail. This bridges the gap between AI prediction and real operational tasks, ensuring the value is realized on the ground, fewer outages, more scheduled maintenance rather than emergencies.

Because the system touches many aspects of operations, governance is important: we could document run-books that map model outputs to actions and responsibilities. For example, a runbook might state: "If predicted RLF probability >80% on any link, then NOC shall verify weather and dispatch field team within 4 hours", effectively creating a new Standard Operating Procedure. Over time, as the model proves its accuracy, more aggressive actions like instant automated fail-over can be allowed under specified conditions, with the run-books updated accordingly.

All automated and manual actions are logged, ensuring complete traceability of what was done, by whom or what, and why. This traceability is critical not only for technical reasons but also for compliance e.g. demonstrating to regulators that any network changes due to AI followed approved policies and did not violate safety or neutrality principles. Finally, by integrating with OSS/BSS, the predictive system contributes to a broader AIOps strategy. It complements other AI use cases such as anomaly detection, capacity planning, and customer experience management [94]. The result is a more proactive operations model for the 5G network, instead of reacting to alarms after a failure which is often too late to avoid impact, the operator can anticipate and prevent outages, improving KPIs like network availability and Mean Time To Restore (MTTR)

[94]. This transformation from reactive to preventive maintenance is a significant value proposition of AI in telecom, as noted by industry analyses [94]. Our deployed GAT-based RLF prediction system is a concrete step in this direction, tightly integrated with existing workflows for maximum practical benefit.

4.5.7.4 Security, Privacy and Compliance

Deploying an AI system in a telecom network demands adherence to strict security and privacy standards. Our RLF prediction framework includes a security-by-design approach to ensure it does not introduce vulnerabilities or compliance issues:

4.5.7.4.1 Data Security All data transitions between network elements, the prediction system, and any cloud components would be encrypted using industry standards. The system interfaces with existing secure NMS databases, and any new data stores are secured with the operator's authentication and access control mechanisms. We would implement role-based access control for the system's components: for example, only authorized service accounts can pull KPI data from the NMS, and only the model service account can access the weather API. The model's predictions might be sensitive as it could reveal vulnerabilities or performance issues, so it needs to be protected and shared only with need-to-know operations personnel or systems.

4.5.7.4.2 Privacy Considerations The data used in our system is mostly network performance and weather data, which generally does not involve personal user information. Therefore, individual privacy concerns like GDPR relating to subscriber data are minimal in this context. We ensure that any potentially sensitive information e.g. site locations, which could be considered confidential infrastructure data is handled in compliance with local regulations: for instance, if processed in the cloud, ensuring data

residency in the required country or region. In the original dataset, precise location data was omitted for security, and in deployment, we would follow the operator's policies on what level of location or identifying info can be used in external systems. Any data sent to third-party services such as a public weather API is carefully vetted to avoid leaking internal network info; typically, only location or time queries are sent outward, not actual network performance data.

4.5.7.4.3 Compliance and standards Telecom operators operate under various standards and regulations (ISO 27001 for information security, NIST frameworks for cyber security, etc.). Our system is designed to comply with these. For example, maintaining audit logs which help fulfill requirements for accountability. If using cloud, we would align with telco guidelines for NFV/Cloud such as leveraging secure VPCs, complying with 3GPP SA5 specifications for management data handling, etc.. We also consider standards like O-RAN Alliance specifications if integrating at the RAN level: ensuring that any edge component using the RAN Intelligent Controller APIs follows the proper interface protocols and security like using the O1/O2 interface for management data as intended, with secure transport.

4.5.7.4.4 Secure Deployment The software components, including the data pipeline and the model service, undergo security hardening to reduce potential vulnerabilities and minimize the attack surface of the system. Security measures include scanning the container image for known vulnerabilities, regular patching of libraries and dependencies (particularly machine learning frameworks), and restricting network exposure by limiting open ports. Any system APIs are not publicly accessible and are restricted to the operator's internal network or cloud environment, with authentication tokens used to control access. In addition, input validation is implemented on incoming data to en-

sure that the pipeline can handle unexpected or malformed inputs gracefully, preventing potential processing errors or instability in downstream components.

4.5.7.4.5 Fault Isolation From a safety standpoint, the predictive system should be kept logically separate from the critical real-time network functions. Even when automating responses, those should go through controlled network management interfaces. This means if the AI system were compromised or malfunctioning, it cannot directly disrupt live traffic. The worst it can do is generate wrong alerts or suggestions, which the operator can override. This isolation is an important design choice to meet the reliability expectations of telecom operations.

4.5.7.4.6 Resilence to Attacks We also consider the scenario of adversarial inputs such as someone feeding false data to trick the model. Weather and KPI sources are assumed to be trusted, coming from internal systems or official meteorological feeds, but if an attacker did spoof metrics to cause false alarms, a multi-layer verification which is operators verifying alerts, multiple data signals needed for major actions should be adopted to provide resilience. Over time, if the model becomes a critical part of operations, further techniques like anomaly detection on the input data sources themselves can be added to detect any intentional tampering. By implementing these security and privacy measures, we ensure the deployment does not compromise the network's integrity.

By adopting such a predictive system, network operators can reduce downtime, enhance reliability, and optimize maintenance costs, thereby improving overall network performance.

4.6 Conclusion

5G radio links are susceptible to failures due to weather impacts. Predicting these failures can reduce downtime, operational cost, and QoS degradation. This research presented a GNN-based model that predicts radio link failures using historical weather data and radio performance metrics, and we showed consistent improvements over traditional machine learning baselines. Given the operational importance of reliable failure prediction in 5G networks, the trade-off between computational cost and accuracy is justified for high-priority network management workflows where preventing failures outweighs inference overhead. Beyond predictive accuracy, we demonstrated deployability considerations relevant to operators, including sub-5-minute ingest-to-alert processing, tunable decision thresholds, and redundancy targets ($\geq$99.9%) aligned with carrier practice. Our ablation results further confirm that principled imbalance handling is essential: without it, the model's utility collapses despite superficially high accuracy driven by majority-class dominance.

Taken together, the results of this chapter show that graph-based learning is valuable not only for recovery and control, but also for proactive reliability prediction in wireless networks. The main thesis-level lesson is that graph representation becomes especially powerful when local network behavior is shaped by neighboring sites, shared environmental exposure, and relational structure that conventional models cannot explicitly capture. Unlike the restoration problem in Chapter 3, this task does not primarily require sequential control, but instead benefits from graph-aware prediction over structured spatial context. This reinforces the broader dissertation claim that GNNs provide a common modeling foundation across different classes of network management problems, even when the downstream objective shifts from action selection to failure anticipation.

Chapter 5

Graph Neural Network-Based Internet Traffic Prediction in 6G Networks with Genetic Algorithm Hyperparameter Optimization

This chapter addresses the third and most anticipatory network management setting considered in this thesis: *proactive planning through forecasting*. In contrast to restoration after failure and prediction of imminent degradation, the focus here is on forecasting future traffic demand in 6G-oriented environments so that network resources can be provisioned more effectively in advance. Within the broader thesis, this chapter examines how graph-based learning can be extended from resilience and reliability tasks into spatio-temporal demand modeling, where both structural dependencies and temporal evolution are central.

5.1 Introduction

The advent of 6G networks brings an explosion of connected devices and data-intensive applications like holographic communications, autonomous systems, VR/AR. These networks must dynamically adapt to rapidly fluctuating traffic loads, making accurate traffic prediction a critical component of network management [98]. Timely forecasts enable proactive resource allocation such network slicing and load balancing to prevent congestion and maintain Quality of Service (QoS) [98]. However, predicting internet traffic in 6G is exceedingly challenging due to its highly dynamic, non-stationary nature [98]. Traffic patterns can shift unpredictably across time and space e.g. across base stations or edge nodes, hence defying the assumptions of simple models and requiring more sophisticated forecasting techniques.

Historically, statistical time-series models like Autoregressive Integrated Moving Average (ARIMA) and Holt-Winters have been used to predict network traffic. These methods can model baseline trends and seasonality and were successfully used for short-term traffic load predictions. However, these methods assume stationarity and linearity, which often fails for modern network traffic characterized by burstiness and non-linear usage patterns [99]. Machine learning approaches such as Support Vector Machines have also been applied [99], but they too struggle to capture the full complexity of traffic dynamics without extensive feature engineering. This has motivated a shift toward data-driven learning-based methods that can automatically discover patterns in vast streams of traffic data.

Deep learning models have achieved remarkable success in time-series forecasting by learning non-linear temporal relationships that traditional methods miss. Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks have been widely adopted for network traffic prediction [100]. LSTMs mitigate the vanishing gradi-

ent problem and can retain long-term memory of traffic trends, enabling more accurate predictions of complex usage patterns than ARIMA-like models [101]. Despite these gains, conventional deep sequence models typically treat each traffic source or link in isolation and do not explicitly leverage correlations between traffic at different network nodes. This limits their ability to capture spatial dependencies. As 6G networks involve highly interconnected infrastructure, ignoring the spatial aspect of traffic can lead to sub-optimal predictions. To address the spatial dimension, Graph Neural Networks (GNNs) have emerged as powerful tools for traffic forecasting on networked data. By modeling the communication network as a graph with nodes such as base stations or routers and edges representing traffic flows or proximity, GNN-based models capture the spatial dependencies and topology influences on traffic [99]. Recent studies in transportation and communication networks show that GNNs achieve state-of-the-art performance in traffic prediction by learning both temporal and spatial patterns simultaneously [99]. In essence, GNN architectures can aggregate information from neighboring nodes, enabling the predictor to understand how traffic at one node impacts others. This capability has led to significant accuracy improvements over traditional approaches; for instance, GNN models have outperformed baseline methods like historical averages and ARIMA in real-world traffic flow datasets [99]. GNN-based spatio-temporal models are now a preferred method for precise traffic forecasts [102], especially in scenarios where relationships in the data are defined by an underlying network graph. While RNNs such as LSTMs capture short and medium-term dependencies, they can struggle with very long-range temporal patterns due to limitations in memory length and the propagation of gradients [103].

The Transformer architecture, with its self-attention mechanism, has recently revolutionized sequence learning by effectively modeling long-range dependencies and complex periodic behaviors in time-series data. In the traffic forecasting domain, Transformer-

based models have begun to challenge the dominance of pure GNN-RNN approaches, often achieving superior accuracy by attending to global time patterns [102]. Notably, the strengths of GNNs and Transformers are complementary: GNNs excel at spatial feature extraction, while Transformers excel at temporal sequence modeling. Combining these can lead to a holistic spatio-temporal learning framework. Early attempts at hybrid models confirm that integrating graph neural layers with Transformer encoders yields improved performance, as the model can jointly learn network topology influences and long-term temporal trends [102]. Such a GNN-Transformer synergy enables capturing both local traffic interactions and global traffic evolution. For 6G traffic, which exhibits intricate spatial correlations and non-periodic bursts over time, a GNN+Transformer approach is promising to achieve more accurate and robust forecasts than either component alone.

Building high-performing deep learning models for traffic prediction requires tuning numerous hyperparameters – learning rates, number of layers, hidden units, activation functions, etc. These design choices critically affect model convergence and generalization. Manually searching for the best configuration is time-consuming and often suboptimal, given the large hyperparameter search space in deep neural networks [104]. Genetic Algorithms offer an efficient, automated way to optimize hyperparameters by mimicking natural selection. Prior research has demonstrated that GA-based hyperparameter tuning can outperform grid search and random search, finding configurations that yield lower forecasting errors [100]. In the context of this study, using a GA is motivated by the need to simultaneously tune multiple components like graph learning rate, attention heads, hidden dimension etc. in the proposed GNN-Transformer model. By leveraging a GA, we aim to efficiently discover a well-tuned model that might otherwise require prohibitive trial-and-error effort.

Despite the advancements discussed above, there is a notable gap in the literature at the intersection of GNN-based traffic prediction and Transformer-driven sequence modeling within 6G networks. Existing works have largely treated these innovations separately: some focus on graph neural nets for network traffic, others on Transformer models, and a few on hyperparameter optimization techniques. No prior study, to our knowledge, has unified GNN and Transformer approaches for 6G traffic prediction while also employing evolutionary hyperparameter optimization. By combining GNNs and Transformers, we target both spatial and temporal complexities of 6G traffic, and by using a Genetic Algorithm for tuning, we seek to unlock the model's full potential without manual intervention. This integrated approach is expected to provide more accurate and reliable traffic forecasts, directly addressing 6G network needs such as real-time adaptive resource allocation, anomaly detection, and proactive congestion avoidance.

In this chapter, we present a Graph Neural Network-Based Internet Traffic Prediction framework for 6G Networks with Genetic Algorithm Hyperparameter Optimization. The key contributions of our study are summarized as follows:

- We develop a new deep learning architecture that integrates Graph Neural Networks with Transformer-based encoders to jointly capture the spatial dependencies and long-range temporal patterns in 6G network traffic. This hybrid model exploits the non-Euclidean structure of communication networks and the sequence modeling power of Transformers, providing a more expressive forecasting capability than traditional RNN or CNN models alone [102].

- We introduce a GA-driven hyperparameter optimization strategy for our GNN-Transformer model. The GA automatically searches for optimal hyperparameters eliminating tedious manual tuning. This approach enhances the model's performance, as evidenced by significant error reductions compared to default or ran-

domly tuned models [100].

- We evaluate the proposed framework on a 6G network traffic scenario, demonstrating its effectiveness in handling the non-stationary, high-dimensional traffic patterns of next-generation networks. Through extensive experiments, we show that our GNN-Transformer model with GA optimization outperforms several baseline methods including classical time-series models (ARIMA/Holt-Winters), pure LSTM/RNN models, and recent GNN-only approaches in forecasting accuracy and stability. The results confirm that combining spatial graph learning with temporal self-attention leads to better generalization in unpredictable 6G traffic environments.

5.2 Related Works

This section reviews the prior work most directly relevant to traffic forecasting in 5G/6G networks. It complements the cross-cutting survey in Chapter 2, focusing on the forecasting methods that motivate the spatio-temporal GNN framework developed in this chapter. In this section, we will review research work using deep learning-based methods to predict traffic.

Wang et al. [105] introduced a time-series similarity-based graph attention network (TSGAN) for spatial-temporal cellular traffic prediction, aiming to improve the accuracy of traffic forecasts in 5G and beyond networks. The model integrates spatial data from the real-world network topology with temporal traffic data using graph neural networks (GNNs). TSGAN utilizes dynamic time warping (DTW) to measure time-series similarity, constructing an adjacency matrix that reflects spatial relationships between cells. This matrix is then combined with traffic data to enhance the model's ability to

predict future traffic demands across both time and space. Through comparisons with several existing models, including GNN-based and GRU-based approaches, the authors showed that TSGAN outperformed the baselines in short-term, mid-term, and long-term prediction scenarios using a real-world cellular traffic dataset.

Mehrabian et al. [106] proposed a dynamic Bernstein graph recurrent network (DBGRN) for wireless cellular traffic prediction, addressing the challenges of modeling spatial-temporal dependencies in large-scale networks. Their approach introduces a spectral dynamic graph construction (SDGC) method that dynamically infers spatial dependencies between cells using a self-attention mechanism and discrete Fourier transform (DFT). To capture complex spatial correlations, they developed a dynamic Bernstein polynomial filtering (DBPF) module, which improves on traditional graph spectral filtering by learning more diverse spectral filters. The DBGRN integrates this filtering module with a gated recurrent unit (GRU) network to predict both spatial and temporal traffic patterns. The authors evaluated their model using a real-world dataset from Telecom Italia and demonstrated significant improvements in prediction accuracy compared to four state-of-the-art baseline models. Their results highlight the effectiveness of incorporating spectral, spatial, and temporal information in wireless traffic forecasting.

Kong et al. in [107] introduced a pure Transformer model for network traffic time-series and reported improved accuracy for long-horizon forecasting compared to recurrent models. The Transformer's attention mechanism could learn periodic patterns (e.g. daily peaks) more visually and effectively than traditional approaches.

Another notable model is Informer [108], a Transformer variant optimized for long time-series forecasting with reduced complexity. Informer achieves high efficiency and accuracy on long sequences by using sparse self-attention and was shown to handle network traffic data with long-term patterns (e.g. weekly cycles) better than LSTM or

CNN-based models.

Wu et al. [109] introduced a Spatio-Temporal Graph Convolutional Recurrent Network (SGCRN) for internet traffic, where each node is a point in a backbone network. SGCRN integrates graph convolution for spatial pattern extraction with a recurrent structure for time series modeling. It showed strong results on real network traffic datasets, accurately predicting traffic matrices (i.e. traffic between many origin-destination pairs) by considering the network's adjacency matrix. Another model, named MSTNN, leveraged graph attention mechanisms for multi-step network traffic forecasting and similarly reported better accuracy than non-graph deep learning models.

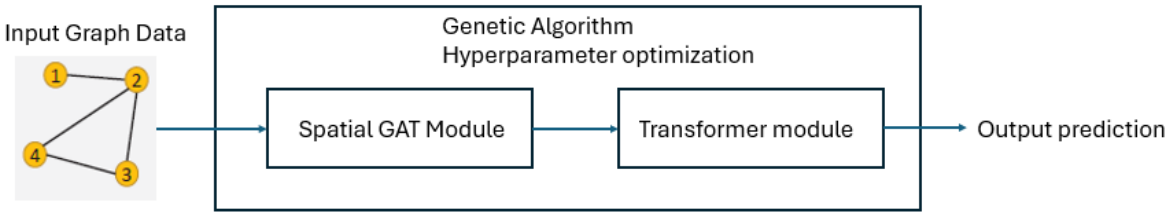


Figure 5.1: Simplified project model

5.3 Methodology

5.3.1 Problem Formulation

We formalize network traffic prediction as a supervised spatio-temporal forecasting problem on a graph. Let the traffic network be represented as a graph $G = (V, E)$, where V is the set of N nodes, i.e cells, and E is the set of edges capturing connectivity between nodes. Each node $i \in V$ is associated with a time series of traffic measurements $x_{i,t}$ (flow) observed at discrete time intervals t. We denote by $\mathbf{X}_t \in \mathbb{R}^{N \times F}$ the vector of all node features at time t, where F is the number of features per node (for single-

feature data, $F = 1$ and $\mathbf{X}_t$ is an N-dimensional vector). The goal is to predict future traffic states for all nodes using historical observations. Given the past T time steps $\mathbf{X}_{t-T+1}, \mathbf{X}_{t-T+2}, \ldots, \mathbf{X}_t$, we aim to learn a function $f(\cdot)$ that outputs a prediction for the next H time steps at all nodes:

$$\hat{X}_{t+1:t+H} = f(X_{t-T+1}, \ldots, X_t, \Theta) \tag{5.1}$$

where Θ is the learnable model parameters, and $\hat{\mathbf{X}}_{t+1:t+H} = [\hat{\mathbf{X}}_{t+1}, \hat{\mathbf{X}}_{t+2}, \ldots, \hat{\mathbf{X}}_{t+H}]$ is the sequence of predicted traffic metrics for all N nodes over the forecast horizon H. This formulation captures the inherent spatial-temporal dependencies in traffic data: traffic conditions vary over time and are influenced by neighboring locations. The spatial aspect reflects the network connectivity, while the temporal aspect reflects the time series dynamics. The learning objective is to train the model f to accurately forecast future traffic. We define a loss function that penalizes the deviation of predictions from ground truth, the mean squared error (MSE) between predicted and actual values over all nodes and forecast times:

$$\mathcal{L}(\Theta) = \frac{1}{HN} \sum_{\tau=1}^{H} \sum_{i=1}^{N} (\hat{x}_{i,t+\tau} - x_{i,t+\tau})^2 \tag{5.2}$$

where $x_{i,t+\tau}$ is the true value and $\hat{x}_{i,t+\tau}$ is the predicted value for node i at time $t + \tau$. The model parameters Θ are optimized to minimize $\mathcal{L}$ over the training data. This formulation encapsulates our goal of learning a mapping from historical spatio-temporal data to future network-wide traffic states.

5.3.2 Graph Representation of Traffic Data

To capture spatial relationships in the traffic network, we structure the data as a graph. Nodes in the graph correspond to traffic cell towers and edges represent spatial connectivity between nodes. In many cases, an adjacency matrix $A \in \mathbb{R}^{N \times N}$ is used to encode the graph structure, where entry A_{ij} indicates the strength of connection from node j to node i. A larger A_{ij} implies that traffic conditions at node i are strongly influenced by node j. If the graph is undirected, A is symmetric. In this work, we construct a sparse K-nearest neighbor (KNN) graph to represent node connectivity. Each node is linked only to its K most similar nodes based on average historical traffic features. Specifically, we compute a feature vector for each node, its average traffic pattern, and dynamically build the adjacency matrix by connecting each node to its K nearest neighbors in this feature space. This KNN-based approach yields a data-driven adjacency matrix that captures meaningful relationships while remaining sparse. By focusing on each node's connections on the most related neighbors, we preserve local spatial correlations and filter out noise from distant unrelated nodes. In the adjacency matrix, we set $A_{ij} = 1$ if node j is among the K nearest neighbors of i, and $A_{ij} = 0$ otherwise. By structuring the data as a graph with adjacency matrix A, we explicitly represent the spatial dependency topology. This graph G serves as a backbone for our spatial modeling: it will be used by the graph neural network component to aggregate information from neighboring nodes.

5.3.3 Spatio-Temporal Model Architecture

Our proposed model integrates a Graph Attention Network (GAT) for learning spatial features on the traffic graph and a Transformer for modeling temporal dynamics. By combining these components, the model can capture complex spatio-temporal dependencies in traffic data. An overview of the architecture is as follows: for each time step in the

input sequence, a GAT module processes the graph of node features to extract latent spatial features; these per-time-step graph embeddings are then fed into a Transformer module that learns temporal patterns across the sequence; finally, a prediction layer uses the transformer's output to produce the forecast for future traffic. The simplified architecture is illustrated in Figure 5.1.

5.3.3.1 Graph Attention Network for Spatial Feature Learning:

Graph Attention Networks (GAT) are a type of GNN that leverage an attention mechanism to learn the importance of each node's neighbors when aggregating information [110][111]. In our model, at each time step t, a GAT layer takes as input the features of all nodes (e.g. the traffic $x_i(t)$ for each node i) and produces updated node embeddings that incorporate information from their neighbors. We denote the feature vector of node i at time t as $\mathbf{h}_i(t) \in \mathbb{R}^F$ (for the first layer, this could be just the scalar $x_i(t)$ or include additional features, so $F = 1$ in a simple case). The GAT layer computes a transformed feature for each node i by attending to its neighbors $j \in \mathcal{N}(i)$. Specifically, the layer uses a trainable attention function $a(\cdot, \cdot)$ to compute attention coefficients:

$$e_{ij}(t) = a\left(Wh_i(t), Wh_j(t)\right) \tag{5.3}$$

for each edge i–j where W is a weight matrix applied to node features. These raw coefficients e_{ij} represent the relevance of node j's features to node i. We then normalize them across all neighbors of i using a softmax:

$$\alpha_{ij}(t) = \frac{\exp(e_{ij}(t))}{\sum_{k \in \mathcal{N}(i)} \exp(e_{ik}(t))} \tag{5.4}$$

so that $\sum_{j \in \mathcal{N}(i)} \alpha_{ij}(t) = 1$. The normalized weight $\alpha_{ij}(t)$ can be interpreted as the attention weight of node j's information when updating node i. Finally, each node's new representation $\mathbf{h}'_i(t)$ is computed as a weighted sum of its neighbors' transformed features, passed through a nonlinear activation $\sigma(\cdot)$:

$$h'_i(t) = \sigma\left(\sum_{j \in \mathcal{N}(i)} \alpha_{ij}(t) W h_j(t)\right) \tag{5.5}$$

Intuitively, node i looks at all other neighbor nodes j and selectively aggregates their traffic information, with larger α_{ij} meaning node j's traffic had more influence on node i's representation at this time. This allows the model to learn spatial dependencies; for example, if two cells often experience correlated spikes, the attention mechanism can assign high weight between those nodes. We can stack multiple GAT layers to capture higher-order neighborhood information or use multi-head attention to stabilize learning. In multi-head GAT, K attention heads compute independent attentions α^k_{ij}, and their outputs are concatenated or averaged to form the final representation [111]. In summary, the GAT module produces an embedding vector for each node at time t that encodes the influence of other nodes' traffic at the same time t. We denote the output of the GAT layer at time t as $\mathbf{z}(t) = [\mathbf{z}_1(t), \ldots, \mathbf{z}_N(t)]$, where $\mathbf{z}_i(t) = \mathbf{h}'_i(t)$ is node i's spatially-informed feature. These $\mathbf{z}_i(t)$ will serve as inputs to the temporal model.

5.3.3.2 Transformer for Temporal Sequence Modeling

To capture the spatial and temporal dynamics of traffic, we utilize a Transformer model to process the sequence of graph features output by the Graph Attention Network (GAT). The Transformer [112] is a sequence-to-sequence model based on self-attention mechanisms, particularly effective for learning long-range dependencies in time-series data. In

our context, the input to the Transformer consists of a sequence of spatial embeddings, denoted as $\mathbf{H}_{t-T+1}, \mathbf{H}_{t-T+2}, \ldots, \mathbf{H}_t$, where each $\mathbf{H}_{t'} \in \mathbb{R}^{N \times D}$ represents the traffic state of the entire network at time step t'. We flatten each $\mathbf{H}_{t'}$ into a single feature vector for each time step, which is then processed for all nodes independently, as the Transformer models all nodes in parallel.

The Transformer encoder takes as input the sequence of vectors $\{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_T\}$, where each $\mathbf{z}_t$ represents the spatial feature vector at time step t (e.g., the flattened $\mathbf{H}_t$). Since the self-attention mechanism in the Transformer is position-agnostic, we add a **positional encoding** to each $\mathbf{z}_t$ to capture the temporal order. This encoding can either be sinusoidal or learned positional embeddings, as proposed in the original Transformer architecture. The positional encoding allows the model to distinguish between traffic states at different times (for example, from 5 hours ago versus 5 minutes ago), thus preserving the temporal order of the data [113].

The core of the Transformer is the multi-head self-attention layer. For each time step t in the encoder, self-attention allows the model to attend to traffic features at other time steps (key positions) and learn temporal correlations [114]. The self-attention mechanism computes a weighted sum of values from all time positions, with the weights determined by the pairwise similarity of the queries and keys. The self-attention output for all time positions is given by:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{5.6}$$

where Q, K, and $V \in \mathbb{R}^{T \times d_k}$ are the query, key, and value matrices, respectively, and d_k is the dimension of the queries and keys. The output from the attention heads is then concatenated and passed through a linear transformation to produce the final attention output.

The attention mechanism allows the model to focus on relevant past time steps for each node, effectively capturing temporal dependencies. The model uses multiple attention heads to enable the learning of diverse features at each time step. These temporal dynamics are learned in conjunction with spatial dependencies, as the input to the Transformer comes from the output of the GAT, which processes the graph to capture spatial correlations between different nodes (i.e., cells in the traffic network). To optimize the model further, we introduce a memory buffer that stores learned information across time steps. After processing the sequence through the Transformer, the memory buffer is updated by blending the current output with the previous memory state. This mechanism ensures that important long-term dependencies are preserved over time. The memory update is computed as:

$$\text{Memory}_{\text{new}} = (1 - \alpha) \cdot \text{Memory}_{\text{old}} + \alpha \cdot \text{Transformer Output} \tag{5.7}$$

where α is the memory update rate, and $\text{Memory}_{\text{old}}$ and Transformer Output are the previous memory state and the current output from the Transformer, respectively.

Finally, the updated memory is combined with the Transformer output, and the result is passed through a fully connected (fc) layer to produce the final prediction for the traffic volume. This output represents the predicted traffic for the entire network at the current time step. In our spatio-temporal model, the Transformer processes the sequence of graph-augmented features, thereby focusing on when past traffic states are informative for predicting the future. By using the GAT and Transformer in tandem, we achieve a separation of concerns: the GAT focuses on spatial relations at each time step, and the Transformer focuses on temporal relations across time steps. This design is aligned with recent advances that integrate graph neural networks with Transformers for traffic forecasting [115].

5.3.3.3 Integrated Spatio-Temporal Model and Prediction Layer:

The GAT and Transformer components are coupled to form a unified spatio-temporal prediction model. Specifically, for each time step in the input window $t - T + 1$ to t, we first apply the GAT (spatial encoder) to compute node embeddings $\mathbf{H}_{t'} = \text{GAT}(\mathbf{X}_{t'}, A)$ for each t' in the sequence. This yields a sequence of graph-structured latent features $\mathbf{H}_{t-T+1}, \ldots, \mathbf{H}_t$, where $\mathbf{H}_{t'} \in \mathbb{R}^{N \times D}$ contains the D-dimensional features for all N nodes at time t'. Next, this sequence is passed to the Transformer (temporal encoder), which processes it as described in the previous section. The Transformer outputs a sequence of context-aware representations $\mathbf{Z}_{t-T+1}, \ldots \mathbf{Z}_t$, where each $\mathbf{Z}_{t'}$ encapsulates the information of the entire historical window as seen from time t''s perspective. We can use the transformer's output in various ways to generate predictions for future time steps:

- Direct one-step prediction: A straightforward approach is to take the Transformer's output corresponding to the latest input time t (which we expect to contain information about the recent history) and map it to a prediction for time $t + 1$. For example, we can use a fully connected prediction layer (a linear projection) that takes $\mathbf{Z}_t$ and produces an output vector $\hat{\mathbf{X}}_{t+1} \in \mathbb{R}^{N \times F}$. This linear layer can be applied node-wise if $\mathbf{Z}_t$ retains node-wise structure, or globally if $\mathbf{Z}_t$ is flattened. In a simple case, if $\mathbf{Z}_t$ is of dimension $N \times D$, the prediction layer could consist of a weight matrix $W_{\text{out}} \in \mathbb{R}^{D \times F}$ (and optionally a bias) such that $\hat{\mathbf{X}}_{t+1} = \mathbf{Z}_t W_{\text{out}}$. This yields the forecast for each node at the next time step.

- Sequence prediction (multi-step): To predict multiple future steps ($H > 1$), one approach is to use the Transformer in a sequence-to-sequence manner. This could involve a Transformer decoder that takes the encoder outputs (from times $t-T+1$ to t) and autoregressively generates outputs for $t + 1$ to $t + H$. Another approach is to extend the output layer to predict H steps in one shot (for instance, having

the output dimension be $N \times F \times H$). For simplicity, our description assumes a one-step prediction which can be extended or iterated for multi-step forecasting.

The final prediction layer thus produces the network's traffic forecast given the transformer's encoded representation of the recent history. The model is trained end-to-end: the GAT and Transformer parameters, as well as the output layer, are optimized together to minimize the loss between $\hat{\mathbf{X}}_{t+1:t+H}$ and the true future $\mathbf{X}_{t+1:t+H}$ (using the loss $\mathcal{L}$ defined in Section 3.1). We use backpropagation through time and space (the gradients flow through both the temporal and spatial components). In training, techniques such as teacher forcing (for multi-step decoders), dropout, and layer normalization help regularize the model. We optimize the model using stochastic gradient descent (Adam optimizer) on the training set and validate on a separate set to avoid overfitting. The choice of hyperparameters (learning rate, batch size, number of GAT heads K, number of Transformer layers L, etc.) is made based on validation performance. In summary, our spatio-temporal model operates in two stages for each prediction: (1) Spatial encoding with a GAT that computes a rich embedding of the traffic graph at each time step, learning which neighboring roads are most relevant for each sensor's state; (2) Temporal encoding with a Transformer that learns how traffic patterns evolve over time, identifying important time dependencies and trends. These components are seamlessly integrated so that the spatial encoder feeds into the temporal encoder. The outcome is a robust model that can learn both local and long-range dependencies in traffic data. The integration of GAT and Transformer has the flexibility to capture complex patterns (e.g., sudden events propagating through the network or periodic congestion cycles), which is essential for accurate traffic forecasting [115].

5.3.4 Genetic Algorithm-Based Hyperparameter Optimization

In machine learning, hyperparameter tuning plays a critical role in determining the performance of a model. For the GAT-Transformer network, the selection of hyperparameters such as the number of GAT heads, hidden dimensions, and learning rate significantly influences its predictive accuracy. Instead of manually selecting these values or performing an exhaustive grid search, we employ a GA to automate and optimize this process. The Genetic Algorithm is inspired by natural selection, where a population of potential solutions evolves over time, with the best solutions being selected and combined to produce even better solutions [116].

In our approach, we define a search space consisting of the hyperparameters that are relevant to the supervised model. The search space includes values for the hidden dimension, learning rate, and number of GAT heads. These hyperparameters are treated as genes within an individual chromosome, which represents a potential solution for the model's configuration. For example, each hyperparameter will be randomly assigned a value from its corresponding list in the search space, creating a random individual.

The optimization process starts by generating an initial population of individuals. Each individual consists of a specific combination of hyperparameters. These individuals are evaluated based on their fitness, which in this case is the model's performance on a validation set, using the validation loss as a metric. Lower validation loss corresponds to a higher fitness score.

Next, we apply crossover and mutation to generate new individuals for the next generation. The crossover function combines two parent solutions to create an offspring with a mix of both parents' hyperparameters. Mutation introduces small random changes to an individual, allowing for the exploration of new hyperparameter combinations that may not have been explored by the crossover. The process of selection, crossover, and

mutation is repeated for a number of generations. In each generation, only the best individuals survive to the next round of evolution, ensuring that the population's fitness improves over time.

After running the genetic algorithm for the specified number of generations, the model with the best-performing hyperparameters is identified. These optimized hyperparameters are then used to train the final model on the entire dataset, and the performance is evaluated on the test set.

The choice of a genetic algorithm over other hyperparameter search strategies is deliberate. Grid search scales exponentially with the number of hyperparameters and quickly becomes intractable, while random search, though more efficient, provides no mechanism for exploiting information from previously evaluated configurations. Bayesian optimization does exploit such information, but it is most effective for smooth, low-dimensional, continuous search spaces and is less natural when the space mixes discrete and continuous hyperparameters, as it does here: the number of GAT heads and the hidden dimension are discrete, while the learning rate is continuous. The objective is also non-differentiable with respect to these hyperparameters and expensive to evaluate, which rules out gradient-based tuning. A genetic algorithm is well matched to this setting: it is a gradient-free, population-based method that naturally handles mixed discrete and continuous variables, explores several regions of the search space in parallel, and refines promising configurations through selection, crossover, and mutation without assuming any structure in the objective. These properties make it a practical and robust choice for tuning the GAT-Transformer, while remaining straightforward to implement and parallelize.

5.4 Evaluation

We apply our proposed model to a real-world mobile traffic dataset provided by Telecom Italia [117] for the city of Milan. The dataset spans from November 1, 2013, to January 1, 2014, with data collected in 10-minute intervals. The city is divided into 10,000 cells, and the dataset includes three types of cellular traffic services: short message service (SMS), call service, and internet service. Since many cells have zero traffic volume in certain intervals, we aggregate the traffic data into hourly intervals for better analysis. To evaluate the performance of our prediction models, we use the mean absolute error (MAE) and root mean squared error (RMSE) as performance metrics.

For our evaluation, we select a 10x10 cell range in the center of Milan and extract December cellular traffic CDRs as time-series data. The first 25 days are used for training, while the remaining 6 days are reserved for testing. The time window length is set to seven time steps, and we use a sliding window method to generate additional training samples.

In the case of the GAT-Transformer model, the adjacency matrix is generated using K-nearest neighbors (KNN). The model runs a GA to find the optimal hyperparameters, which are then used to retrain the final model on the entire dataset. Based on the GA results, the optimal hyperparameters are determined to be a hidden layer size of 64, a learning rate of 0.005, 4 attention heads, and 2 encoder layers. The Adam optimizer is used for model training. We compare the performance of our proposed prediction model with the following baseline models:

- LSTM [118], which uses multiple LSTMs to predict mobile traffic.
- DBGRN [106] which uses a dynamic Bernstein graph current network for traffic prediction.

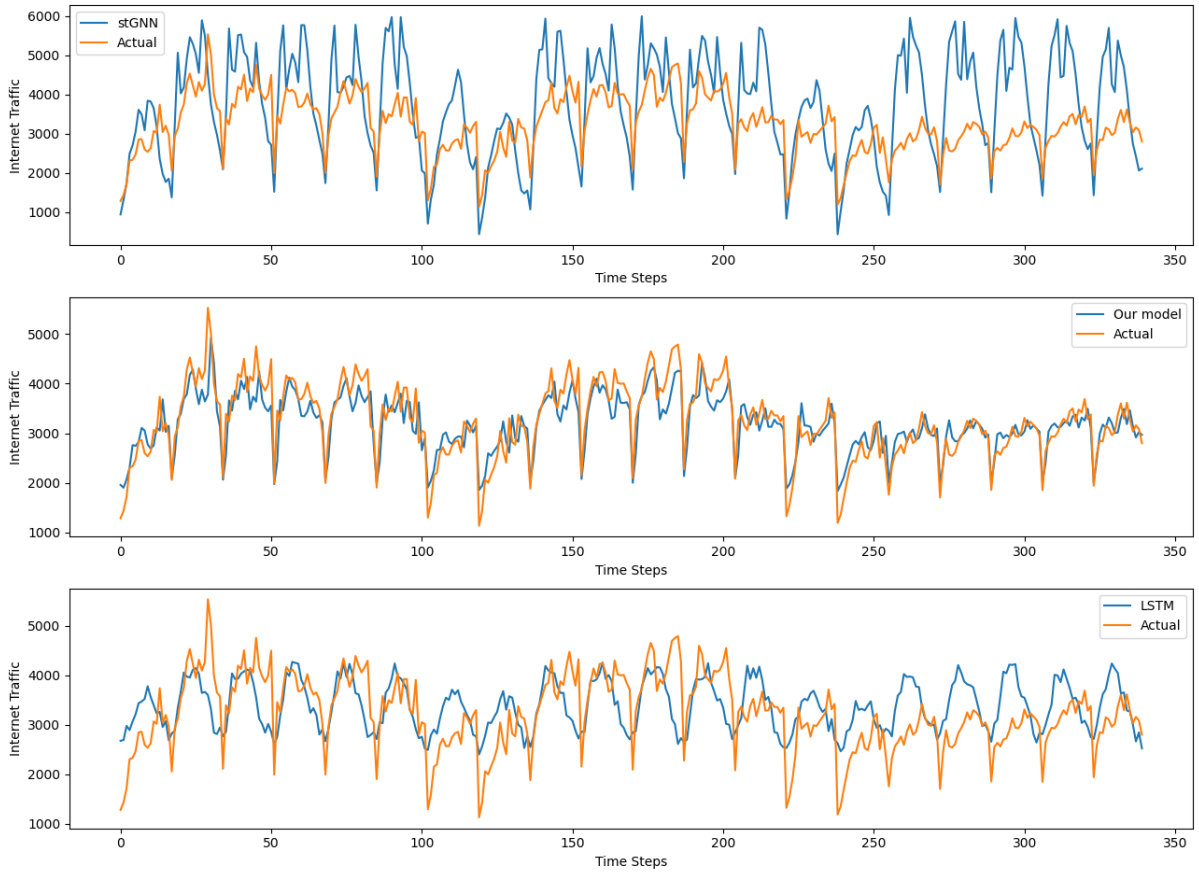


Figure 5.2: Actual versus predicted internet traffic for our model and two baseline models (stGNN and LSTM) for the same cell tower. The vertical axis reports the raw aggregated activity value from the Telecom Italia Milan dataset, a dimensionless measure of cellular activity.

- stGNN [119], which integrates graph Fourier transform, DFT, and deep neural networks to capture both inter- and intra-series dependencies in multivariate time series forecasting.

According to the results in Table 5.1, the GAT-Transformer model outperforms the other models in terms of both RMSE (Root Mean Squared Error) and MAE (Mean Absolute Error). GAT-Transformer achieves an RMSE of 125.9, which is the lowest among all models. DBGRN follows with an RMSE of 181.7, showing a 27.1% higher error than the GAT-Transformer. stGNN has an RMSE of 233.7, which is 85.6% higher than the GAT-

Transformer. LSTM performs better than stGNN, with an RMSE of 146.5, representing a 16.4% increase over the GAT-Transformer. The GAT-Transformer model exhibits a substantial improvement in RMSE compared to the other models. This indicates that the GAT-Transformer is significantly more accurate in predicting future traffic states, with lower deviation from the true values. GAT-Transformer also achieves an MAE of 93.5, the best among all models. DBGRN follows with an MAE of 121.6, which is 29.9% higher than the GAT-Transformer. stGNN has an MAE of 126.8, representing a 35.5% increase compared to the GAT-Transformer. LSTM performs better than stGNN, with an MAE of 125.2, which is 33.9% higher than the GAT-Transformer. Similarly to RMSE, the GAT-Transformer model demonstrates the best MAE performance, with the lowest error in terms of absolute differences between predicted and actual values. This reinforces the advantage of GAT-Transformer in minimizing prediction errors compared to the baseline models.

The GAT-Transformer model consistently outperforms the other models (DBGRN, stGNN, and LSTM) across both RMSE and MAE metrics. This indicates that the GAT-Transformer is better at capturing the spatial and temporal dependencies inherent in the traffic data, leading to more accurate predictions. The attention mechanism in the GAT helps the model focus on the most relevant features and spatial relations between nodes, while the Transformer component models long-term temporal dependencies, contributing to improved prediction accuracy.

These performance improvements demonstrate the effectiveness of integrating graph-based attention mechanisms and temporal sequence modeling for traffic prediction, making the GAT-Transformer a superior choice for such forecasting tasks.

Figure 5.2 shows a comparison of actual vs predicted traffic for a single cell tower using three different models: stGNN, our model and LSTM. The Top Plot shows stGNN

Table 5.1: Comparison of model prediction performance

Model	RMSE	MAE
GAT-Transformer(our model)	125.9	93.5
DBGRN	181.7	121.6
stGNN	233.7	126.8
LSTM	146.5	125.2

- Actual vs Predicted Traffic. The stGNN model (shown in blue) does not capture the actual traffic (orange line) very well, especially in high traffic peaks and troughs. While it follows the general trend of the traffic, there are noticeable discrepancies, particularly in sudden rises or drops in traffic. The model fails to accurately predict the traffic in these critical areas, and the overall prediction is more smoothed out compared to the actual values.

The Middle Plot shows our model - Actual vs Predicted Traffic. The GAT-Transformer model (shown in blue) demonstrates a much closer fit to the actual traffic (orange line). It tracks the traffic more precisely, closely following the peaks and valleys of the actual traffic data. This shows that the model effectively captures both the spatial and temporal dependencies in the traffic data, resulting in more accurate predictions. While there are minor deviations, especially in the more stable traffic periods, the model outperforms stGNN significantly.

The LSTM model (shown in blue) in the bottom performs better than stGNN but still lags behind the GAT-Transformer. The LSTM captures the general trend of the traffic well but struggles to match the peaks and sharp drops in traffic.

The GAT-Transformer model provides the most accurate predictions of internet traffic for this particular cell tower, outshining both stGNN and LSTM in predicting fluctua-

tions and overall traffic patterns. This highlights the effectiveness of combining graph-based attention mechanisms for spatial dependencies and transformer layers for temporal modeling in traffic prediction tasks.

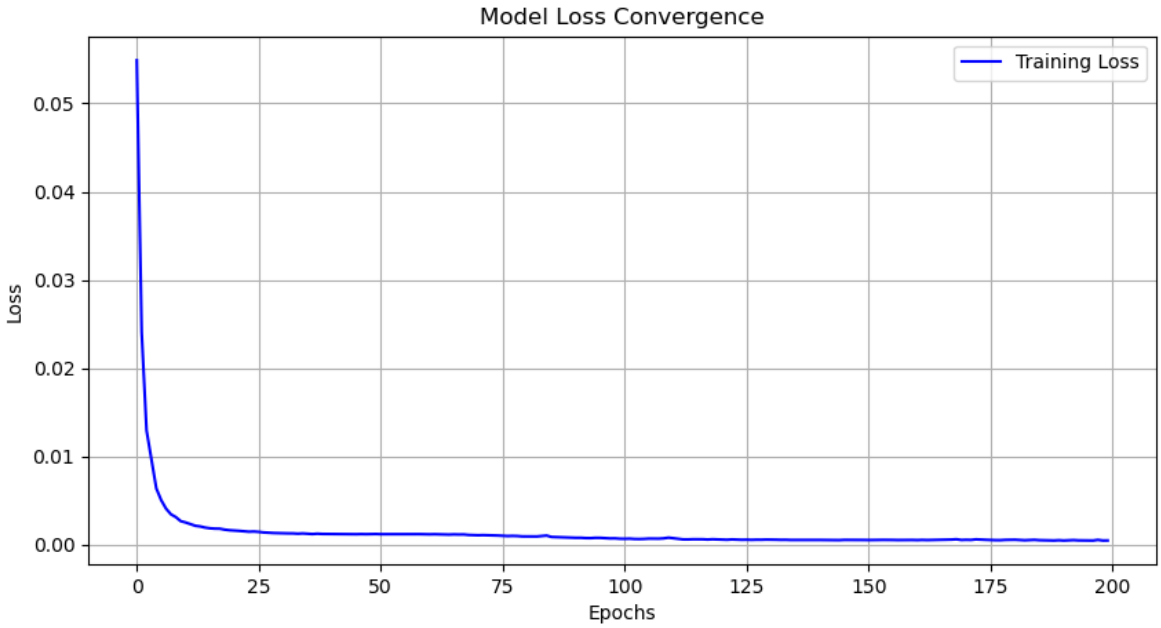


Figure 5.3: Convergence performance of our model

5.5 Conclusion

This chapter introduced a novel approach for predicting internet traffic in 6G networks using a hybrid model that integrates Graph Attention Networks (GAT) and Transformer encoders. The model effectively captures both spatial dependencies and long-range temporal dynamics. Additionally, GA-based hyperparameter optimization is used to fine-tune the model, improving its predictive accuracy. Experimental results show that the GAT-Transformer model outperforms other state-of-the-art models, such as DBGRN,

stGNN, and LSTM, with the lowest RMSE of 125.9 and MAE of 93.5, demonstrating its superior performance in dynamic 6G environments.

The GAT-Transformer model benefits from its ability to focus on the most relevant spatial neighbors through the attention mechanism in GAT, while the Transformer effectively captures long-term traffic trends and sudden fluctuations. This makes the GAT-Transformer highly effective in addressing the complex spatio-temporal nature of 6G traffic, a task where traditional models often struggle. Also, the integration of GA for hyperparameter tuning ensures that the model configuration is optimized for the best performance, eliminating the need for exhaustive manual search. This process has proven to be efficient and effective, contributing to the model's high predictive accuracy and generalizability across different traffic scenarios.

In conclusion, our proposed GAT-Transformer model provides a robust and scalable solution for internet traffic prediction in 6G networks. By combining graph-based spatial learning with temporal modeling and automated hyperparameter optimization, this approach is well-positioned to support critical applications in 6G, such as real-time resource allocation, congestion management, and Quality of Service (QoS) maintenance.

This chapter extends the dissertation's central argument into the domain of predictive resource planning. The results show that graph-based learning remains effective when the management objective shifts from failure handling to future demand estimation, particularly in settings where traffic patterns are shaped by both network structure and temporal dynamics. At the same time, the chapter highlights an important boundary condition of graph representation: for forecasting tasks, spatial structure must be complemented by temporal modeling and configuration optimization to achieve strong performance. This supports the broader thesis conclusion that GNNs serve as the unifying structural backbone across intelligent network management tasks, while complementary mechanisms

such as sequence models and genetic optimization become necessary when the problem is strongly temporal and model-sensitive.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

This thesis investigated the role of Graph Neural Networks (GNNs) in intelligent network management across three representative communication network problems: restoration in optical WDM networks, radio link failure prediction in 5G networks, and traffic forecasting in 6G-oriented environments. While these problems differ in application domain, timescale, and operational objective, the dissertation has shown that they share a common structural foundation: they all arise in communication systems whose behavior is governed by graph-structured dependencies, evolving network state, and uncertainty. The central outcome of this thesis is therefore not only the development of three task-specific models, but the demonstration that graph-based learning provides a coherent and effective foundation for intelligent network management across heterogeneous networking contexts.

Taken together, the three studies show that network restoration, failure prediction, and traffic forecasting can be understood as three complementary manifestations of the same broader challenge: enabling communication networks to act more adaptively under

disruption, uncertainty, and growth. In the WDM restoration problem, the objective is reactive and control-oriented: the system must respond quickly to failure and make effective path and wavelength decisions. In the radio link failure prediction problem, the objective is predictive: the system must detect risk before service degradation occurs. In the traffic forecasting problem, the objective is anticipatory: the system must infer future demand so that capacity and resources can be allocated proactively. Although these tasks differ operationally, this thesis has shown that they all benefit from modeling the network as a structured relational system rather than as a collection of isolated observations.

A key conclusion of the dissertation is that graph representation helps most when network entities cannot be treated independently. In all three studies, the underlying topology, spatial dependency, or interaction structure among network elements carried information that conventional models or heuristic approaches could not fully exploit. In optical restoration, graph-based state representation improved the ability to reason over topology-aware recovery choices. In radio link failure prediction, graph construction enabled the model to incorporate shared weather and site relationships that affect reliability. In traffic forecasting, graph-based spatial modeling improved the ability to capture interdependencies across flows and network elements that shape future demand. Across these domains, the dissertation demonstrates that the principal value of GNNs lies in their ability to convert topological dependence into learnable structure.

At the same time, the thesis also shows that graph representation alone is not always sufficient. The three studies indicate that graph-based learning is most powerful when paired with additional mechanisms suited to the nature of the task. In restoration, the problem is inherently sequential and decision-oriented; therefore, GNN-based state encoding must be complemented by Deep Reinforcement Learning (DRL) to support action

selection under evolving network conditions. In failure prediction and traffic forecasting, temporal dependence becomes central; thus, graph-based spatial learning must be combined with temporal modeling mechanisms to capture how network behavior evolves over time. In the forecasting study, Genetic Algorithm (GA)-based hyperparameter optimization further improves the practical effectiveness of the learning framework by adapting the model configuration to the demands of the forecasting task. These findings suggest that GNNs should be viewed not as a universal standalone solution, but as a unifying representational backbone that can be strengthened by task-specific learning and optimization techniques.

More broadly, the dissertation demonstrates a progression from reactive to predictive to anticipatory network management. The restoration study addresses how networks recover after disruption. The failure prediction study addresses how networks anticipate degradation before disruption. The forecasting study addresses how networks anticipate future demand before performance bottlenecks emerge. This progression is important because it reflects the larger shift in modern communication systems away from static, rule-based management and toward adaptive, learning-driven control. In that sense, the thesis contributes not only three individual technical studies, but also a broader perspective on how graph-based intelligence can support increasingly autonomous forms of network management.

6.1.1 Insights

Several broader insights emerge from the three studies.

First, network topology is not merely context; it is part of the learning signal. Traditional approaches often rely on handcrafted features, localized rules, or flattened representations of network state. The work in this thesis shows that such approaches can

miss important dependencies between links, sites, and traffic flows. By explicitly modeling relational structure, graph-based learning enables richer representations of network behavior and improves adaptability across tasks.

Second, the usefulness of graph-based learning increases with system complexity and coupling. As the number of interactions among network components grows, the limitations of independent-link or independent-flow models become more pronounced. The results across optical restoration, 5G reliability prediction, and 6G traffic forecasting suggest that graph-based models are especially valuable in environments where local decisions depend on broader network state.

Third, intelligent network management requires combining representation learning with task-aware decision mechanisms. GNNs effectively capture structure, but the downstream task still matters. Sequential control requires reinforcement learning. Long-range temporal dependence benefits from sequence modeling such as Transformers. Hyperparameter and architecture sensitivity may require search and optimization methods such as GA. A major lesson of this thesis is therefore that the future of intelligent network management lies not in a single monolithic model, but in principled combinations of graph-based representation with problem-specific inference or control strategies.

Fourth, practical relevance depends on more than predictive performance alone. Across the thesis, runtime, scalability, generalization, and deployment considerations emerged as critical factors. This is particularly important because communication networks are operational systems, not only benchmark environments. The value of graph-based intelligence in this context lies not only in accuracy gains, but in its potential to support real-time or operationally meaningful decisions.

6.1.2 Scope and Boundaries of the Thesis

It is important to clarify the scope of the conclusions drawn in this dissertation.

This thesis does not claim that a single model architecture can solve all network management problems. Nor does it claim that graph-based learning should replace all traditional methods in communication networks. Instead, the thesis establishes that graph-based learning provides a strong and flexible foundation for problems in which network structure, interdependence, and dynamic state are central to decision quality.

This thesis has several limitations that should be acknowledged. First, the studies are evaluated in specific network settings and datasets, and performance may vary across operators, vendors, regions, frequency bands, and evolving deployment conditions. Second, some tasks considered in the thesis, particularly radio link failure prediction, involve rare-event labels derived from operational records, which may introduce ambiguity and label noise. Third, some graph constructions rely on practical surrogates for structural or environmental dependence, such as weather-station proximity, which may not fully capture all relevant exogenous factors or localized variations. Fourth, some of the proposed frameworks are evaluated offline, meaning that real-world deployment would still require threshold calibration, drift monitoring, periodic retraining, and operational validation under live conditions. Accordingly, the contribution of this thesis is best understood as establishing and validating a graph-centered methodological foundation for intelligent network management, rather than claiming a complete or universally deployable autonomous network management solution.

6.1.3 Future Work

Several directions emerge naturally from this work. For restoration problems, future research should investigate improved scalability through hierarchical learning, graph sam-

pling, transfer learning, and parallelized training, as well as broader resilience objectives such as simultaneous node and link failures and energy-aware restoration strategies. For predictive reliability management, future work should focus on multi-region validation, domain adaptation across heterogeneous networks, improved labeling with higher-resolution timestamps and root-cause information, and the integration of additional operational signals such as alarms, trouble tickets, maintenance records, and configuration changes. It would also be valuable to incorporate calibrated uncertainty estimation, risk-aware decision support, and online deployment capabilities such as drift detection and periodic retraining. More broadly, the graph-based reliability framework could be extended beyond weather-driven failures to other degradation modes, including hardware aging and equipment faults, through multi-cause or multi-task learning formulations. These directions would further strengthen the role of graph-based learning as a foundation for adaptive, predictive, and operationally relevant network management.

References

[1] gokhankosem. Global Mobile Data Traffic Forecast 2027 * IpCisco — ipcisco.com. https://ipcisco.com/global-mobile-data-traffic-forecast-2027, 2025. [Accessed 03-06-2025].

[2] Wagdy M Othman, Abdelhamied A Ateya, Mohamed E Nasr, Ammar Muthanna, Mohammed ElAffendi, Andrey Koucheryavy, and Azhar A Hamdi. Key enabling technologies for 6g: The role of uavs, terahertz communication, and intelligent reconfigurable surfaces in shaping the future of wireless networks. *Journal of Sensor and Actuator Networks*, 14(2):30, 2025.

[3] Kazi Hasan, Thomas Trappenberg, and Israat Haque. A generalized transformer-based radio link failure prediction framework in 5g rans. *arXiv preprint arXiv:2407.05197*, 2024.

[4] Weiwei Jiang, Jiayun Luo, Miao He, and Weixi Gu. Graph neural network for traffic forecasting: The research progress. *ISPRS International Journal of Geo-Information*, 12(3):100, 2023.

[5] Prohim Tam, Seyha Ros, Inseok Song, Seungwoo Kang, and Seokhoon Kim. A survey of intelligent end-to-end networking solutions: Integrating graph neural networks and deep reinforcement learning approaches. *Electronics*, 13(5):994, 2024.

[6] Robin Matzner, Ruijie Luo, Georgios Zervas, and Polina Bayvel. Intelligent performance inference: A graph neural network approach to modeling maximum achievable throughput in optical networks. *APL Machine Learning*, 1(2), 2023.

[7] Usama Mehboob, Junaid Qadir, Salman Ali, and Athanasios Vasilakos. Genetic algorithms in wireless networking: techniques, applications, and issues. *Soft Computing*, 20:2467–2501, 2016.

[8] Olivier Crochat, J-Y Le Boudec, and Ornan Gerstel. Protection interoperability for wdm optical networks. *IEEE/ACM transactions on networking*, 8(3):384–395, 2000.

[9] VV Dhanya, Santhos Kumar Das, and Sarat Kumar Patra. Qos based light path provisioning and performance analysis in wdm network. In *2012 International Conference on Computing, Electronics and Electrical Technologies (ICCEET)*, pages 659–662. IEEE, 2012.

[10] Admin. Technologies, 2025. URL https://www.3gpp.org/technologies/5g-system-overview#:~:text=The%205GC%20architecture%20relies%20on,approach%20offers%20modularity%20and%20reusability.

[11] Sabine Dahmen-Lhuissier. 5g, 2025. URL https://www.etsi.org/technologies/5g?tmpl=component#:~:text=ITU,R%20M.2083.

[12] H Pennanen, T Hänninen, O Tervo, A Tölli, and M Latva-aho. 6g: The intelligent network of everything—a comprehensive vision, survey, and tutorial. arxiv 2024. *arXiv preprint arXiv:2407.09398*, 2024.

[13] Francine Cássia de Oliveira, Gustavo Iervolino de Morais, João Victor Menino e Silva, and Ramon Magalhães Nogueira. Use cases and 6g architecture: New

needs and challenges. *ICN 2023 : The Twenty-Second International Conference on Networks*, 2023.

[14] Karim Boutiba, Miloud Bagaa, and Adlen Ksentini. Radio link failure prediction in 5g networks. In *2021 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6. IEEE, 2021.

[15] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.

[16] William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[17] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

[18] Yu Xiong, Yuanyuan Li, Bin Zhou, Ruyan Wang, and George N Rouskas. Sdn enabled restoration with triggered precomputation in elastic optical inter-datacenter networks. *Journal of Optical Communications and Networking*, 10(1):24–34, 2017.

[19] S Mohd Sam, S Mohd Daud, Kamilia Kamardin, and Nurazean Maarop. Study of qos performance in optical burst switched networks (obs). *Indian Journal of Science and Technology*, 9:48, 2016.

[20] Neeraj Mohan, Surbhi Gupta, and Abhay Bhandari. Link failure recovery using p-cycles in wavelength division multiplex (wdm) mesh networks. *Journal of Optical Communications*, 44(s1):s937–s944, 2024.

[21] Masahiko Jinno, Tomohiko Takagi, and Yuto Uemura. Enhanced survivability of translucent elastic optical network employing shared protection with fallback. In *Optical Fiber Communication Conference*, pages Th3K–1. Optica Publishing Group, 2017.

[22] Amit Wason and RS Kaler. Fault-tolerant routing and wavelength assignment algorithm for multiple link failures in wavelength-routed all-optical wdm networks. *Optik*, 122(2):110–113, 2011.

[23] Anwar Alyatama. Adaptive spectrum allocation algorithm for elastic optical networks with survivability. In *2018 International Conference on Computing Sciences and Engineering (ICCSE)*, pages 1–6. IEEE, 2018.

[24] Yuanhao He, Geyang Xiao, Jun Zhu, Tao Zou, and Yuan Liang. Reinforcement learning-based sdn routing scheme empowered by causality detection and gnn. *Frontiers in Computational Neuroscience*, 18:1393025, 2024.

[25] HW Chong, Sam Kwong, and Kim-Fung Man. Optimization of spare capacity in wdm networks for real-time service restoration. *Applied Artificial Intelligence*, 20 (8):639–654, 2006.

[26] Rujia Zou, Nathaniel Bury, Hiroshi Hasegawa, Masahiko Jinno, and Suresh Subramaniam. Deepdrama: Deep reinforcement learning-based disaster recovery with mitigation awareness in eons. In *2021 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6. IEEE, 2021.

[27] Yulong Ma, Yingya Guo, Ruiyu Yang, and Huan Luo. Frrl: A reinforcement learning approach for link failure recovery in a hybrid sdn. *Journal of Network and Computer Applications*, 234:104054, 2025.

[28] Michael Doherty, Robin Matzner, Rasoul Sadeghi, Polina Bayvel, and Alejandra Beghelli. Reinforcement learning for dynamic resource allocation in optical networks: Hype or hope? *arXiv preprint arXiv:2502.12804*, 2025.

[29] Shyamal Krishna Agarwal, Somesh Banerjee, and Rajarshi Mahapatra. Prediction and recovery of radio link failure caused by environmental factors. In *2022 IEEE 19th India Council International Conference (INDICON)*, pages 1–6. IEEE, 2022.

[30] Semih Aktaş, Hande Alemdar, and Salih Ergüt. Towards 5g and beyond radio link diagnosis: Radio link failure prediction by using historical weather, link parameters. *Computers and Electrical Engineering*, 99:107742, 2022.

[31] Mahmoud Said Elsayed, Nhien-An Le-Khac, Soumyabrata Dev, and Anca Delia Jurcut. Network anomaly detection using lstm based autoencoder. In *Proceedings of the 16th ACM symposium on QoS and security for wireless and mobile networks*, pages 37–45, 2020.

[32] Irina Kochetkova, Anna Kushchazli, Sofia Burtseva, and Andrey Gorshenin. Short-term mobile network traffic forecasting using seasonal arima and holt-winters models. *Future Internet*, 15(9):290, 2023.

[33] Fengli Xu, Yuyun Lin, Jiaxin Huang, Di Wu, Hongzhi Shi, Jeungeun Song, and Yong Li. Big data driven mobile traffic understanding and forecasting: A time series approach. *IEEE transactions on services computing*, 9(5):796–805, 2016.

[34] Tao Ma, Constantinos Antoniou, and Tomer Toledo. Hybrid machine learning algorithm and statistical time series model for network-wide traffic forecast. *Transportation Research Part C: Emerging Technologies*, 111:352–372, 2020.

[35] Qu Zhaowei, Li Haitao, Li Zhihui, and Zhong Tao. Short-term traffic flow forecasting method with mb-lstm hybrid network. *IEEE Transactions on Intelligent Transportation Systems*, 23(1):225–235, 2020.

[36] Sajal Saha, Saikat Das, and Glaucio HS Carvalho. Convlstmtransnet: A hybrid deep learning approach for internet traffic telemetry. *arXiv preprint arXiv:2409.13179*, 2024.

[37] Qingsong Wen, Tian Zhou, Chaoli Zhang, Weiqi Chen, Ziqing Ma, Junchi Yan, and Liang Sun. Transformers in time series: A survey. *arXiv preprint arXiv:2202.07125*, 2022.

[38] Biswanath Mukherjee. *Optical WDM networks*. Springer Science & Business Media, 2006.

[39] Michael To and Philippe Neusy. Unavailability analysis of long-haul networks. *IEEE journal on selected areas in communications*, 12(1):100–109, 1994.

[40] Kejie Lu, Jason P Jue, Timucin Ozugur, Gaoxi Xiao, and Imrich Chlamtac. Intermediate-node initiated reservation (iir): a new signaling scheme for wavelength-routed networks with sparse conversion. In *IEEE International Conference on Communications, 2003. ICC'03.*, volume 2, pages 1386–1390. IEEE, 2003.

[41] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80, 2008.

[42] Zhuoran Li, Xing Wang, Ling Pan, Lin Zhu, Zhendong Wang, Junlan Feng, Chao

Deng, and Longbo Huang. Network topology optimization via deep reinforcement learning. *IEEE Transactions on Communications*, 71(5):2847–2859, 2023.

[43] Peyman Kafaei, Quentin Cappart, Nicolas Chapados, Hamed Pouya, and Louis-Martin Rousseau. Dynamic routing and wavelength assignment with reinforcement learning. *INFORMS Journal on Optimization*, 6(1):1–18, 2024.

[44] Nicola Di Cicco, Emre Furkan Mercan, Oleg Karandin, Omran Ayoub, Sebastian Troia, Francesco Musumeci, and Massimo Tornatore. On deep reinforcement learning for static routing and wavelength assignment. *IEEE Journal of Selected Topics in Quantum Electronics*, 28(4: Mach. Learn. in Photon. Commun. and Meas. Syst.): 1–12, 2022.

[45] Sam Nallaperuma, Zelin Gan, Josh Nevin, Mykyta Shevchenko, and Seb J Savory. Interpreting multi-objective reinforcement learning for routing and wavelength assignment in optical networks. *Journal of Optical Communications and Networking*, 15(8):497–506, 2023.

[46] Xiangyu Zheng, Wanwei Huang, Hui Li, and Guangyuan Li. Research on generalized intelligent routing technology based on graph neural network. *Electronics*, 11 (18):2952, 2022.

[47] Mohamed Mostafa A Azim, Xiaohong Jiang, Pin-Han Ho, Md MR Khandker, and Susumu Horiguchi. A new scheme for lightpath restoration in wdm networks. In *Workshops on Mobile and Wireless Networking/High Performance Scientific, Engineering Computing/Network Design and Architecture/Optical Networks Control and Management/Ad Hoc and Sensor Networks/Compil*, pages 381–386. IEEE, 2004.

[48] Kaiyuan Zhao, Zinan Zhao, Zhenyong Wang, and Hongjiang Zhang. Routing algorithm design based on deep reinforcement learning and gnn. In *IEEE INFOCOM 2024-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 1–6. IEEE, 2024.

[49] Isaac Ampratwum and Amiya Nayak. Optimizing wdm network restoration with deep reinforcement learning and graph neural networks integration. In *2024 IEEE 48th Annual Computers, Software, and Applications Conference (COMPSAC)*, pages 794–800. IEEE, 2024.

[50] Quentin Cappart, Didier Chételat, Elias B Khalil, Andrea Lodi, Christopher Morris, and Petar Veličković. Combinatorial optimization and reasoning with graph neural networks. *Journal of Machine Learning Research*, 24(130):1–61, 2023.

[51] Richard S Sutton and Andrew G Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2 edition, 2018.

[52] Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8: 279–292, 1992.

[53] Volodymyr Mnih. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[54] Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.

[55] LC Freeman. A set of measures of centrality based on betweenness. *Sociometry*, 1977.

[56] Paul Almasan, José Suárez-Varela, Krzysztof Rusek, Pere Barlet-Ros, and Albert Cabellos-Aparicio. Deep reinforcement learning meets graph neural networks: Exploring a routing optimization use case. *Computer Communications*, 196:184–194, 2022.

[57] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. {TensorFlow}: a system for {Large-Scale} machine learning. In *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, pages 265–283, 2016.

[58] G Brockman. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

[59] Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[60] Fernando Barreto, Emílio CG Wille, and Luiz Nacamura Jr. Fast emergency paths schema to overcome transient link failures in ospf routing. *arXiv preprint arXiv:1204.2465*, 2012.

[61] Naser Al-Falahy and Omar Y. K. Alani. Millimetre wave frequency band as a candidate spectrum for 5g network architecture: A survey. *Physical Communication*, 32:120–144, 2019.

[62] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009.

[63] Jacek Rak. A new approach to design of weather disruption-tolerant wireless mesh networks. *Telecommunication Systems*, 61:311–323, 2016.

[64] Massimo Tornatore et al. A survey on network resiliency methodologies against weather-based disruptions. In *Proceedings of the 8th International Workshop on Resilient Networks Design and Modeling (RNDM)*, pages 23–34, 2016.

[65] Forough Yaghoubi, Marija Furdek, Ahmad Rostami, Peter Öhlén, and Lena Wosinska. Design and reliability performance of wireless backhaul networks under weather-induced correlated failures. *IEEE Transactions on Reliability*, 71(2):616–629, 2022.

[66] Ibraheem Shayea, Liyth A Nissirat, Mahdi A Nisirat, Aida Alsamawi, Tharek A Rahman, Marwan H Azmi, Mohammad Abo-Zeed, and Issam Trrad. Rain attenuation and worst month statistics verification and modeling for 5g radio link system at 26 ghz in malaysia. *Transactions on Emerging Telecommunications Technologies*, 30(12):1–19, 2019.

[67] Semih Aktaş, Hande Alemdar, and Salih Ergüt. Towards 5g and beyond radio link diagnosis: Radio link failure prediction by using historical weather, link parameters. *Computers and Electrical Engineering*, 99:107742, 2022.

[68] Ying Du, Yadong Liu, Xuhong Wang, Jian Fang, Gehao Sheng, and Xiuchen Jiang. Predicting weather-related failure risk in distribution systems using bayesian neural network. *IEEE Transactions on Smart Grid*, 12(1):350–360, 2020.

[69] Mohammad Ariful Islam, Hisham Siddique, Wenbin Zhang, and Israat Haque. A deep neural network-based communication failure prediction scheme in 5g ran. *IEEE Transactions on Network and Service Management*, 20(2):1140–1152, 2023.

[70] Semih Aktaş, Hande Alemdar, and Salih Ergüt. Towards 5g and beyond radio link

diagnosis: Radio link failure prediction by using historical weather, link parameters. *Computers and Electrical Engineering*, 99:107742, 2022.

[71] R Govindasamy, S. K Nagarajan, J. R Muthu, and M Ramkumar. Residual multi-scale attention based modulated convolutional neural network for radio link failure prediction in 5g. *Ad Hoc Networks*, 166:103679, 2025.

[72] K Hasan, K Papry, T Trappenberg, and I Haque. A generalized gnn-transformer-based radio link failure prediction framework in 5g ran. *IEEE Transactions on Machine Learning in Communications and Networking*, 3:710–724, 2025.

[73] Md. R Palas, Md. R Islam, P Roy, Md. A Razzaque, A Alsanad, S. A AlQahtani, and M. M Hassan. Multi-criteria handover mobility management in 5g cellular network. *Computer Communications*, 174:81–91, 2021.

[74] Karim Boutiba, Miloud Bagaa, and Adlen Ksentini. Radio link failure prediction in 5g networks. In *Proceedings of the IEEE Global Communications Conference (GLOBECOM)*, Madrid, Spain, 2021.

[75] Z Zhao, G Verma, C Rao, A Swami, and S Segarra. Link scheduling using graph neural networks. *IEEE Transactions on Wireless Communications*, 22(6):3397–4012, 2023.

[76] J.-H. Huh, T. Inagaki, J. Nakazato, M. Arai, K. Yano, and M. Hasegawa. Graph neural network-based multi-metric performance modeling in urban multi-rat wireless networks. *ICT Express*, 11(5):957–962, 2025.

[77] A Hasan, C Boeira, K Papry, Y Ju, Z Zhu, and I Haque. Root cause analysis of anomalies in 5g ran using graph neural network and transformer. *arXiv preprint arXiv:2406.15638*, 2024.

[78] B. A. Gonzalez, M. Farreras, M. Groshev, J. A. Trujillo, I. de la Bandera, and R. Barco. Graph neural networks for o-ran mobility management: A link prediction approach. *IEEE Vehicular Technology Magazine*, 20(4):2–11, 2025.

[79] Rex Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. Gnnexplainer: Generating explanations for graph neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, Vancouver, Canada, 2019.

[80] J He, A Rafiey, G Mishne, and Y Wang. Explaining gnn explanations with edge gradients. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 884–895, 2025.

[81] R Kubo and D Difallah. Xgexplainer: Robust evaluation-based explanation for graph neural networks. In *Proceedings of the 2024 SIAM International Conference on Data Mining (SDM)*, pages 64–72, 2024.

[82] Antonio Longa, Serena Azzolin, Gabriele Santin, Giulia Cencetti, Pietro Liò, Bruno Lepri, and Andrea Passerini. Explaining the explainers in graph neural networks: a comparative study. *ACM Computing Surveys*, 57(5):1–37, 2025.

[83] Chao Wu, Fan Wu, Lingjuan Lyu, Tao Qi, Yong Huang, and Xing Xie. A federated graph neural network framework for privacy-preserving personalization. *Nature Communications*, 13:3091, 2022.

[84] H. N. Doan, T. N. Xuan, Q. V. Do, H. W. Joo, C. Sang-Hwa, and K. Jong-Deok. Graph neural network-based federated learning for sum-rate maximization in small-cell wireless network. In *Proceedings of the 12th International Symposium on Information and Communication Technology*, pages 243–247, Ho Chi Minh City, Vietnam, 2023.

[85] Z Tan, G Wan, W Huang, and M Ye. Fedssp: Federated graph learning with spectral knowledge and personalized preference. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 34561–34581, Vancouver, Canada, 2024.

[86] Turkcell. Itu-aimlin5gchallenge-2021. https://github.com/Turkcell/ITU-AIMLin5GChallenge-2021, 2025. Accessed: 2025-11-06.

[87] Gerald W Buetow Jr, Ronald Sellers, Donald Trotter, Elaine Hunt, and Willie A Whipple Jr. The benefits of rebalancing. *Journal of Portfolio Management*, 28(2): 23, 2002.

[88] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, Vancouver, Canada, 2018.

[89] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[90] Xiang-Yang Wang, Hong-Ying Yang, Yong-Wei Li, Wei-Yi Li, and Jing-Wei Chen. A new svm-based active feedback scheme for image retrieval. *Engineering Applications of Artificial Intelligence*, 37:43–53, 2015.

[91] Jair Cervantes, Farid Garcia-Lamont, Lisbeth Rodríguez-Mazahua, and Asdrubal Lopez. A comprehensive survey on support vector machine classification: Applications, challenges and trends. *Neurocomputing*, 408:189–215, 2020.

[92] Zhiyong Yang, Taohong Zhang, Jingcheng Lu, Dezheng Zhang, and Dorothy Kalui. Optimizing area under the roc curve via extreme learning machines. *Knowledge-Based Systems*, 130:74–89, 2017.

[93] M Altalhan, A Algarni, and M Turki-Hadj Alouane. Imbalanced data problem in machine learning: A review. *IEEE Access*, 13:13686–13699, 2025.

[94] Avenga. Ai in the telecom industry: preventing network failures. https://www.avenga.com/magazine/ai-telecom/, 2025. Accessed: 2025-11-06.

[95] Ericsson. Ericsson network manager. https://www.ericsson.com/en/network-management, 2025. Accessed: 2025-11-06.

[96] Nokia. Network management system. https://documentation.nokia.com/nsp/24-4/Glossary/ai8g3n88af.html, 2025. Accessed: 2025-11-06.

[97] Isaac Ampratwum and Amiya Nayak. Radio link failure prediction in 5g networks using graph neural networks. In *Proceedings of the IEEE Annual Computers, Software, and Applications Conference (COMPSAC)*, pages 729–736, Toronto, Canada, 2025.

[98] C. Hu, H. Zhou, D. Wu, X. Chen, J. Yan, and X. Liu. Self-refined generative foundation models for wireless traffic prediction. *arXiv preprint*, arXiv:2408.10390, August 2024.

[99] Petros Brimos, Areti Karamanou, Evangelos Kalampokis, and Konstantinos Tarabanis. Graph neural networks and open-government data to forecast traffic flow. *Information*, 14(4):228–228, April 2023.

[100] C. Erden. Genetic algorithm-based hyperparameter optimization of deep learning models for pm2.5 time-series prediction. *International Journal of Environmental Science and Technology*, 20(3):2959–2982, January 2023. doi: 10.1007/s13762-023-04763-6.

[101] T. Shelatkar, S. Tondale, S. Yadav, and S. Ahir. Web traffic time series forecasting using arima and lstm rnn. In *ITM Web of Conferences*, volume 32, page 03017, 2020. doi: 10.1051/itmconf/20203203017.

[102] H. Wang et al. Stgformer: Efficient spatiotemporal graph transformer for traffic forecasting. *arXiv preprint*, arXiv:2410.00385, October 2024. doi: 10.48550/arxiv.2410.00385.

[103] Y. Li and F. Moura. Forecaster: A graph transformer for forecasting spatial and time-dependent data. *arXiv preprint*, arXiv:1909.04019, January 2019. doi: 10.48550/arxiv.1909.04019.

[104] X. Xiao, M. Yan, S. Basodi, C. Ji, and Y. Pan. Efficient hyperparameter optimization in deep learning using a variable length genetic algorithm. *arXiv preprint*, arXiv:2006.12703, June 2020.

[105] Z. Wang, J. Hu, G. Min, Z. Zhao, Z. Chang, and Z. Wang. Spatial-temporal cellular traffic prediction for 5g and beyond: A graph neural networks-based approach. *IEEE Transactions on Industrial Informatics*, 19(4):5722–5731, April 2023. doi: 10.1109/TII.2022.3182768.

[106] A. Mehrabian, S. Bahrami, and V. W. S. Wong. A dynamic bernstein graph recurrent network for wireless cellular traffic prediction. In *IEEE International Conference on Communications (ICC)*, May 2023. doi: 10.1109/ICC45041.2023.10279102.

[107] Q. Kong et al. Network traffic prediction: Apply the transformer to time series forecasting. *Mathematical Problems in Engineering*, 2022:1–6, October 2022. doi: 10.1155/2022/8424398.

[108] Haoyi Zhou, Shanshan Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 11106–11115, 2021. doi: 10.1609/aaai.v35i12.17325.

[109] K. He, X. Chen, Q. Wu, S. Yu, and Z. Zhou. Graph attention spatial-temporal network with collaborative global-local learning for citywide mobile traffic prediction. *IEEE Transactions on Mobile Computing*, 21:1244–1256, 2022.

[110] Y. Song, R. Luo, T. Zhou, C. Zhou, and R. Su. Graph attention informer for long-term traffic flow prediction under the impact of sports events. *Sensors*, 24 (15):4796–4796, July 2024. doi: 10.3390/s24154796.

[111] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio. Graph attention networks. *arXiv preprint*, arXiv:1710.10903, October 2017. URL http://arxiv.org/abs/1710.10903.

[112] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[113] J. Zhou, E. Liu, W. Chen, S. Zhong, and Y. Liang. Navigating spatio-temporal heterogeneity: A graph transformer approach for traffic forecasting. *arXiv preprint*, arXiv:2408.10822, 2024.

[114] S. Cristina. The transformer attention mechanism. https://machinelearningmastery.com/the-transformer-attention-mechanism/, October 2021. Accessed: 2025-06-02.

[115] F. Yan and Q. Chen. Transformer-based spatial-temporal graph attention network for traffic flow prediction. In *International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC)*, pages 132–135, 2023. doi: 10.1109/CyberC58899.2023.00031.

[116] M. Mitchell. *An Introduction to Genetic Algorithms*. MIT Press, Cambridge, MA, 1996.

[117] G. Barlacchi, M. De Nadai, R. Larcher, A. Casella, C. Chitic, G. Torrisi, F. Antonelli, A. Vespignani, A. Pentland, and B. Lepri. A multi-source dataset of urban life in the city of milan and the province of trentino. *Scientific Data*, 2:150055, October 2015.

[118] H. D. Trinh, L. Giupponi, and P. Dini. Mobile traffic prediction from raw data using lstm networks. In *IEEE Int. Symp. Personal, Indoor and Mobile Radio Communications (PIMRC)*, September 2018.

[119] D. Cao, Y. Wang, J. Duan, C. Zhang, X. Zhu, C. Huang, Y. Tong, B. Xu, J. Bai, J. Tong, and Q. Zhang. Spectral temporal graph neural network for multivariate time-series forecasting. In *Advances in Neural Information Processing Systems (NeurIPS)*, December 2020.