Enabling Intelligent Network Management through Multi-Agent Systems: An Implementation of Autonomous Network System

Petro Mushidi Tshakwanda

University of New Mexico - Main Campus

Follow this and additional works at: https://digitalrepository.unm.edu/ece_etds

Part of the Computer and Systems Architecture Commons, Data Storage Systems Commons, Digital Communications and Networking Commons, and the Systems and Communications Commons

Recommended Citation

This Dissertation is brought to you for free and open access by the Engineering ETDs at UNM Digital Repository. It has been accepted for inclusion in Electrical and Computer Engineering ETDs by an authorized administrator of UNM Digital Repository. For more information, please contact disc@unm.edu.
Petro Mushidi Tshakwanda

Candidate

Electrical and Computer Engineering

Department

This dissertation is approved, and it is acceptable in quality and form for publication:

Approved by the Dissertation Committee:

Dr. Michael Devetsikiotis
Chair

Dr. Sisay Tadesse Arzo
Member

Dr. Ali Bidram
Member

Dr. Riccardo Bassoli
Member
Enabling Intelligent Network Management through Multi-Agent Systems:
An Implementation of Autonomous Network System

BY

Petro Mushidi Tshakwanda
B.S., Computer Science, University of Kinshasa, 2007
M.S., Information Systems Management, University of Liverpool, 2016
M.S., Financial Engineering, World Quant University, 2019
M.S., Computer Engineering, University of New Mexico, 2023

DISSERTATION

Submitted in Partial Fulfillment
of the Requirements for the Degree of
Doctor of Philosophy

Engineering
The University of New Mexico
Albuquerque, New Mexico

December, 2023
DEDICATION

In humble recognition and gratitude to God, The Lord Jesus Christ, for His grace, guidance, and unwavering presence throughout this journey.

To my beloved parents, Samuel Tshinyama Konga and Martine Kaji Kamwengo, thank you for instilling in me the values of integrity, perseverance, and the pursuit of knowledge. Your unwavering belief in my abilities has been a constant source of inspiration.

Dedicated to my loving wife Nicole Kayowa Tshibuyi, whose unwavering support and understanding have been the pillar of strength throughout this arduous pursuit. Your love, encouragement, and sacrifices have fueled my determination to overcome every challenge.

To my incredible children El Beryith Tshinyama Mushidi, Grace Kamwanya Mushidi, Israel Mushidi Tshakwanda, and David Koj Mushidi, you are my greatest motivation and source of joy. Your boundless love, laughter, and curiosity have constantly reminded me of the wonders and responsibilities of this world.

To my extended family and friends, thank you for your unwavering faith, prayers, and encouragement. Your support has sustained me during the most challenging times, and I am grateful for your presence in my life.

Finally, I dedicate this work to future generations of researchers and computer engineers. May it serve as a stepping stone towards the advancement of knowledge and innovation. Let us unite in pushing the boundaries of what is possible, leveraging technology for the betterment of humanity.
ACKNOWLEDGMENTS

I would like to express my deepest gratitude to my advisors, Dr. Michael Devetsikiotis and Dr. Sisay Tadesse Arzo, for their exceptional guidance, unwavering support, and invaluable mentorship throughout my doctoral journey. Their expertise, insight, and dedication have been instrumental in shaping my research and fostering my growth as a researcher. I am immensely grateful to the members of my dissertation committee, Dr. Ali Bidram, and Dr. Riccardo Bassoli, for their insightful feedback, valuable suggestions, and constructive criticism. Their expertise and contributions have significantly enriched the quality of this dissertation. I extend my appreciation to the faculty and staff of Electrical and Computer Engineering, The University of New Mexico, for providing a stimulating academic environment and the necessary resources for my research. Their commitment to excellence and their passion for knowledge have been inspiring. I would like to thank my colleagues and fellow researchers for their collaboration, thought-provoking discussions, and invaluable insights. Their collective expertise and camaraderie have played a crucial role in shaping my research ideas and broadening my horizons.

My heartfelt thanks go to the organizations, funding agencies, and scholarships that have generously supported my research endeavors throughout my doctoral studies. Their financial support has been essential in enabling me to conduct experiments, attend conferences, and acquire the necessary resources for my research.

I am grateful to my family and friends for their unconditional love, encouragement, and unwavering belief in my abilities. Their constant support, understanding, and motivation have been a source of strength during the challenging phases of this journey. Finally, I would like to express my deepest appreciation to all those individuals who have contributed, directly or indirectly, to the completion of this dissertation. Your collective support, encouragement, and belief in my research have been invaluable.
Enabling Intelligent Network Management through Multi-Agent Systems:
An Implementation of Autonomous Network System

by

Petro Mushidi Tshakwanda

B.S., Computer Science, University of Kinshasa, 2007
M.S., Information Systems Management, University of Liverpool, 2016
M.S., Financial Engineering, World Quant University, 2019
M.S., Computer Engineering, University of New Mexico, 2023
Ph.D., Engineering, University of New Mexico, 2023

ABSTRACT

This Ph.D. dissertation presents a pioneering Multi-Agent System (MAS) approach for intelligent network management, particularly suited for next-generation networks like 5G and 6G. The thesis is segmented into four critical parts. Firstly, it contrasts the benefits of agent-based design over traditional micro-service architectures. Secondly, it elaborates on the implementation of network service agents in Python Agent Development Environment (PADE), employing machine learning and deep learning algorithms for performance evaluation. Thirdly, a new scalable approach, Scalable and Efficient DevOps (SE-DO), is introduced to optimize agent performance in resource-constrained settings. Fourthly, the dissertation delves into Quality of Service (QoS) and Radio Resource Management using reinforcement learning agents. Lastly, an Autonomous, Intelligent AI/ML Framework is proposed for proactive management and dynamic routing in 6G networks, using advanced algorithms like Speed Optimized LSTM. Overall, the work holds substantial promise for transforming network management through automation, adaptability, and advanced intelligence.
# TABLE OF CONTENTS

List of Figures ................................................................. xii  
List of Tables ................................................................. xiv  

1 Introduction ................................................................. 1  
   1.1 Background and motivation ........................................... 1  
   1.2 Research objectives and contributions .............................. 4  
   1.3 Dissertation outline ................................................ 6  

2 Transforming Network Management with Intelligent Approaches .. 11  
   2.1 Introduction .......................................................... 11  
   2.2 Softwarized network architecture .................................. 12  
   2.3 Charting next-generation networks: The Rise of Softwarization and In-Network Intelligence .............................................. 16  
   2.4 Revolutionizing SDN Controllers: Microservices & Multi-Agent Systems ................................................................. 17  
      2.4.1 Microservices-based SDN ...................................... 17  
      2.4.2 Multi-Agent-based SDN ....................................... 18  
      2.4.3 Smart Topology Learning in MSN Framework: Unleashing Intelligent Agents ......................................................... 21  
   2.5 Intelligent Network Management in Next-Generation Networks .. 26  
      2.5.1 Machine Learning for Network Management ................. 27  
      2.5.2 Deep Learning for Anomaly Detection and Security Threat Mitigation ................................................................. 27  
      2.5.3 Reinforcement Learning for Real-Time Network Decisions . 28  
      2.5.4 Hybrid and Ensemble Learning Approaches ................. 29  
      2.5.5 Federated Learning for Decentralized Learning ............ 29  
   2.6 Chapter Summary ..................................................... 30  

vi
### 3 Softwarized Intelligent Network Architecture Design

#### 3.1 Monolithic and Microservice Architectures: A Comparative Study of Benefits and Challenges

3.1.1 Unearthing the Superiority of Multi-Agent Systems over Microservices: An In-depth Examination

3.1.2 Architecting Intelligent Networks for Next-Generation Connectivity: An Exploration

#### 3.2 Advanced Internal Architecture of Agents: A Cognitive Leap

#### 3.3 Elucidating the System Architecture: A Multi-Agent Perspective

#### 3.4 Strategizing Agent Deployment: A Crucial Aspect of Network Management

#### 3.5 Addressing Real-world Challenges through Advanced Agent and System Architecture

3.5.1 Optimizing Network Performance with the Intelligent Network Traffic Classifier Agent

3.5.2 Intelligent Network Traffic Predictor Agent behavior

3.5.3 Performance Optimization via Intelligent QoS Monitoring and Provisioning Agent

#### 3.6 Case Studies: Paradigm Shifts in Network Service Management Leveraging Intelligent Agents

3.6.1 Leveraging PADE’s Potential: Streamlining Multi-Agent System Implementation

3.6.2 Synergizing PADE with MAS Architecture for Advanced Network Management

3.6.3 Towards Optimal Multi-Agent Systems: Comprehensive Evaluation Metrics for Performance and Efficiency

3.6.4 Delving Deeper: A Comprehensive Examination of Case Study Results
3.7 Chapter Summary .......................................................... 57

4 SE-DO Framework: Boosting Intelligent Agent Efficiency in Resource-Limited IoT Networks 59

4.1 Transforming Telecommunication Landscape: Harnessing Multi-Agent Strategies for 6G Networks and Beyond ........................................ 60

4.2 Intelligent Agents in Resource-Constrained Environments: Challenges and Innovative Solutions ...................................................... 61

4.3 SE-DO: Scalable & Adaptive Agent Deployment in Resource-Constrained Environments ............................................................ 63

4.3.1 Dynamic Agent Lifecycle Orchestration for Resource-Constrained Environments ................................................................. 64

4.4 A New Hierarchical Architecture for Smart Network Agent Automation . 66

4.5 Efficient Strategies for Next-Generation Network Optimization .......... 68

4.5.1 Data Collection Agent ...................................................... 68

4.5.2 The Role of the Data Analytic Agent .................................... 74

4.5.3 The Role of the Data Training Agent .................................... 75

4.5.4 The Role of the Data Predictor Agent .................................... 77

4.6 Enhancing AI Performance in Next-Generation Networks: An Explo-
ration of the SE-DO Framework .............................................. 79

4.6.1 Deciphering the Fabric of Network Topology .......................... 80

4.6.2 Dataset ................................................................. 82

4.6.3 Scalable, Resilient, and Efficient: Unveiling the Performance of Multi-Agent Systems in 6G-IoT Networks through the SE-DO Framework .................................................... 83

4.6.4 Scalability Analysis ....................................................... 85
4.6.5 In-depth Analysis of Data Collection Agents: Comparative Assessments on Resource Allocation, Energy Utilization, and Cost Dynamics ................................................................. 86
4.6.6 Communication Delays Analysis between Agents .......................... 88
4.7 Chapter Summary ........................................................................... 90

5 Quality of Service (QoS) and Radio Resource Management (RRM) 93
5.1 Advancing Towards Intelligent Networking: A Proposal for a Proactive QoS Agent in Beyond 5G and 6G Networks ...................................... 94
5.2 AI in Networking: Neural Networks for Traffic Forecasting ................. 95
5.3 Design Principles for Intelligent QoS Agents in Network Management ... 97
5.3.1 Assessing the Impact of Intelligent QoS Agents on Network Efficiency and User Experience ............................................................... 102
5.4 Augmenting Wireless Network Performance: A Comprehensive Exploration of Opportunistic Packet Scheduling Algorithms ................. 106
5.4.1 Unveiling the Proportional Fair Scheduler: A Key Actor in Autonomous Network Systems ................................................................. 108
5.4.2 Generalized Proportional Fair Scheduler: A Fine-tuned Approach for Autonomous Network Systems ......................................................... 109
5.4.3 Adaptive Generalized Proportional Fair Scheduler ......................... 111
5.4.4 Empirical Exploration: Delving into Scheduler Dynamics .................. 115
5.5 Chapter Summary ........................................................................... 118

6.1 Introduction ................................................................................. 121
6.1.1 Background and Need for Enhanced 6G Network Management . . 123
6.1.2 Role of Multi-Agent Systems in 6G Networks .............................. 125
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.2</td>
<td>AI/ML-Agents and Their Paradigm Shift in Network Management</td>
<td>129</td>
</tr>
<tr>
<td>6.2.1</td>
<td>Overview of AI/ML-Agents in Network Management</td>
<td>129</td>
</tr>
<tr>
<td>6.2.2</td>
<td>Integrating AI/ML into BDI Model for Multi-Agent Systems</td>
<td>130</td>
</tr>
<tr>
<td>6.3</td>
<td>SP-LSTM: A Novel Agent in 6G Networks</td>
<td>133</td>
</tr>
<tr>
<td>6.3.1</td>
<td>Introduction to Speed-optimized LSTM (SP-LSTM)</td>
<td>133</td>
</tr>
<tr>
<td>6.3.2</td>
<td>Role and Performance of SP-LSTM in Multi-Agent Systems</td>
<td>138</td>
</tr>
<tr>
<td>6.4</td>
<td>Reinforcement Learning: An Interactive Agent for Dynamic Routing</td>
<td>140</td>
</tr>
<tr>
<td>6.4.1</td>
<td>The Deployment of Reinforcement Learning within the Context of Autonomous Agents</td>
<td>140</td>
</tr>
<tr>
<td>6.4.2</td>
<td>Interaction of Reinforcement Learning with Other Agents in the System</td>
<td>141</td>
</tr>
<tr>
<td>6.5</td>
<td>Pioneering Network Management through Predictive Intelligence and Dynamic Adaptation in Multi-Agent Systems</td>
<td>142</td>
</tr>
<tr>
<td>6.6</td>
<td>Innovating Network Management: A Fusion of Predictive Intelligence and Dynamic Adaptation</td>
<td>144</td>
</tr>
<tr>
<td>6.7</td>
<td>The Potential of Multi-Agent Systems to Outperform Traditional Network Management Techniques</td>
<td>148</td>
</tr>
<tr>
<td>6.7.1</td>
<td>Network Topology</td>
<td>149</td>
</tr>
<tr>
<td>6.7.2</td>
<td>Dataset</td>
<td>151</td>
</tr>
<tr>
<td>6.7.3</td>
<td>Predictive Analytics and Congestion Analysis</td>
<td>152</td>
</tr>
<tr>
<td>6.7.4</td>
<td>AI/ML-Based Strategies for Optimal Network Management</td>
<td>152</td>
</tr>
<tr>
<td>6.7.5</td>
<td>Scalable Prediction of Network Congestion: A Mathematical Perspective</td>
<td>156</td>
</tr>
<tr>
<td>6.7.6</td>
<td>SP-LSTM: A Pioneering Stride in 6G Network Analytics</td>
<td>157</td>
</tr>
<tr>
<td>6.7.7</td>
<td>Analytical Resilience: Decrypting Optimal Network Path and Predicting Congestion</td>
<td>158</td>
</tr>
</tbody>
</table>
6.7.8 Prospects of Multi-Agent Systems in Nurturing Future Network Management ................................................. 161

6.8 Chapter Summary ................................................................................................................................. 164

7 Conclusion and Future Work ................................................................................................................. 167

7.1 Introduction and Epilogue of the Research Journey ........................................................................... 167

7.2 Summary of Research Findings ........................................................................................................ 168

7.3 Contributions to Knowledge ............................................................................................................ 169

7.4 Practical Implications and Industry Relevance ................................................................................. 171

7.5 Reflection on the Challenges, Limitations, and Paths to Overcoming ............................................. 172

7.6 Blueprint for Future Exploration: Recommendations and Aspirations ........................................... 174

7.7 Concluding Remarks ......................................................................................................................... 178

Appendices .................................................................................................................................................. 180

References .................................................................................................................................................... 188
## LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Synergy of SOA, NFV, and SDN for adaptive network services.</td>
<td>13</td>
</tr>
<tr>
<td>2.2</td>
<td>The architecture of Software-Defined Networking (SDN)</td>
<td>15</td>
</tr>
<tr>
<td>2.3</td>
<td>MSN Decomposition Architecture.</td>
<td>18</td>
</tr>
<tr>
<td>2.4</td>
<td>An Overview of the microservice-based SDN Architecture with Intelligent Agents</td>
<td>20</td>
</tr>
<tr>
<td>2.5</td>
<td>Updating shortest path agent via topology learning service.</td>
<td>22</td>
</tr>
<tr>
<td>2.6</td>
<td>Various network topologies</td>
<td>24</td>
</tr>
<tr>
<td>2.7</td>
<td>Communication delay between a pair of nodes in a client-server setup</td>
<td>25</td>
</tr>
<tr>
<td>2.8</td>
<td>Discovery Time for Different Network Topologies</td>
<td>25</td>
</tr>
<tr>
<td>2.9</td>
<td>Results from Experiments on Shortest Path</td>
<td>26</td>
</tr>
<tr>
<td>3.1</td>
<td>Contrasting Monolithic and Microservice Architectures.</td>
<td>34</td>
</tr>
<tr>
<td>3.2</td>
<td>Cognitive-Driven Agent Architecture.</td>
<td>39</td>
</tr>
<tr>
<td>3.3</td>
<td>MANA-NMS simulation based on the multi-agent system. Adapted from [150].</td>
<td>40</td>
</tr>
<tr>
<td>3.4</td>
<td>Agents Chaining. Adapted from [150].</td>
<td>41</td>
</tr>
<tr>
<td>3.5</td>
<td>MAS-PADE framework</td>
<td>51</td>
</tr>
<tr>
<td>3.6</td>
<td>Mean Absolute Error of different models on Validation and Test data sets.</td>
<td>55</td>
</tr>
<tr>
<td>4.1</td>
<td>SE-DO Approach: multi-agent process</td>
<td>64</td>
</tr>
<tr>
<td>4.2</td>
<td>Hierarchical Architecture</td>
<td>66</td>
</tr>
<tr>
<td>4.3</td>
<td>An illustrative depiction of the 6G network experimental setup</td>
<td>82</td>
</tr>
<tr>
<td>4.4</td>
<td>Accuracy comparison between micro-services-based and agent-based</td>
<td>88</td>
</tr>
<tr>
<td>4.5</td>
<td>Performance of Data Collector Agents</td>
<td>89</td>
</tr>
<tr>
<td>4.6</td>
<td>Communication delay via RESTful API (in µs) between different agents.</td>
<td>90</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>--------</td>
<td>------------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>5.1</td>
<td>An undirected graph</td>
<td>97</td>
</tr>
<tr>
<td>5.2</td>
<td>Network Integration with SDN and OSPF</td>
<td>103</td>
</tr>
<tr>
<td>5.3</td>
<td>Comparison of DL-Based-SDN and OSPF</td>
<td>104</td>
</tr>
<tr>
<td>5.4</td>
<td>Performance Comparison of PF, GPF, and AGPF</td>
<td>115</td>
</tr>
<tr>
<td>6.1</td>
<td>The comprehensive 6G network architecture</td>
<td>128</td>
</tr>
<tr>
<td>6.2</td>
<td>Comparison of Basic and Enhanced BDI Models</td>
<td>131</td>
</tr>
<tr>
<td>6.3</td>
<td>AI/ML-Powered Predictive and Dynamic Routing System</td>
<td>147</td>
</tr>
<tr>
<td>6.4</td>
<td>NetSim designed topology</td>
<td>150</td>
</tr>
<tr>
<td>6.5</td>
<td>Comparison of LSTM and SP-LSTM Model Performance</td>
<td>158</td>
</tr>
<tr>
<td>6.6</td>
<td>An illustrative snapshot</td>
<td>161</td>
</tr>
<tr>
<td>6.7</td>
<td>Variation of Sum of Rewards and Length of Path per Episode</td>
<td>162</td>
</tr>
</tbody>
</table>
# LIST OF TABLES

1.1 Dissertation Structure ................................................................. 7

2.1 Comparison of Different Networking and Software Architectures ........ 14
2.2 Overview of ML, DL, and RL Techniques and Applications in Network Management ................................................................. 30

3.1 Comparison of Microservices and MAS. ........................................... 37
3.2 Development of Multi-Agent Systems: A Survey of Platforms. .......... 50
3.3 Effect of Data Set Size on Performance of MAS Using ANN Architecture 53
3.4 Agent Communication Latency in Seconds. ...................................... 54
3.5 Network Throughput Prediction: Accurate Results at 1-Hour Intervals Using LSTM. ................................................................. 56

4.1 Scalability and Resilience Analysis (Micro-services-based Strategy) .... 86
4.2 Scalability and Resilience Analysis (Agent-based Strategy) ............... 87
4.3 Performance of Data collector agent ................................................. 88
4.4 Performance of Data collector agent ................................................. 89
4.5 Communication delay via RESTful API (in µs) between different agents. 90

5.1 LSTM Predicted Throughputs for Links at One-Hour Intervals .......... 105
5.2 Key differences among PF, GPF, and AGPF opportunistic Schedulers. .. 114
5.3 Performance Comparison of PF, GPF, and AGPF .............................. 116

6.1 Principal Attributes of the Speed-Optimized LSTM .......................... 138
6.2 Q-Learning Architecture for Routing Decisions .............................. 146
6.3 Performance Metrics of the SP-LSTM Model ................................... 157
6.4 Comparison between Traditional LSTM and SP-LSTM ...................... 159
6.5 Network links ranked from most to least congested: Before and After Prediction ........................................... 160
6.6 Episodes with Non-Zero Sum of Rewards and Path Length .............. 163
6.7 Q-Learning in Action: Streamlining Resource Utilization through Optimal Path Selection ........................................... 164
6.8 Optimal Path Analysis ........................................... 164
6.9 Alternative Paths Analysis ........................................... 165
This dissertation represents extensive research in Computer Engineering, exploring innovative solutions to complex problems through data analysis, experimental design, and the development of novel methodologies. The objective is to contribute to the advancement of knowledge by addressing significant challenges in Computer Engineering, particularly in the field of intelligent network management in autonomous systems using multi-agent systems.

Key research areas include agent-based modeling, distributed decision-making, and self-organization, which have been approached with an interdisciplinary perspective integrating computer science, electrical engineering, mathematics, and related fields.

This dissertation does not claim to solve all questions or challenges but aims to make a meaningful contribution to the existing body of knowledge, paving the way for further advancements in Computer Engineering.

I would like to express my gratitude to my advisors, committee members, and colleagues for their support and guidance throughout this research journey. Their expertise and insightful discussions have shaped the direction and scope of this work.

I am also grateful to the funding agencies, industry partners, and research institutions for providing the necessary resources and financial support, enabling the successful completion of this research.

Lastly, I would like to acknowledge the unwavering support of my family, friends, and loved ones. Their encouragement has been a constant source of motivation during this challenging endeavor.

It is my hope that this dissertation significantly contributes to the field of Computer Engineering, inspiring future researchers to explore new possibilities and advancements in this ever-evolving discipline.
Chapter 1

Introduction

1.1 Background and motivation

The telecommunications sphere is currently amidst an unprecedented surge in the integration of devices, sensors, and appliances, fostering a complexity in network management and control that is not to be underestimated. In light of this accelerated growth, traditional methodologies in network management are proving insufficient [132]. These approaches, while tried and tested, are fundamentally challenged by their limited capacity to process the colossal amount of data emitted by these intricate and diverse networks.

Indeed, as businesses and industries are driven by the quest for a competitive edge, mere data exchange is no longer enough. Instead, the evolution is geared towards metamorphosing raw data into an advanced platform for disseminating not only information but also invaluable insights and expertise. The implications of this evolutionary trajectory are manifold and profound, not least in the realm of network management.

The prevailing dynamics within the telecommunications industry unequivocally underscore the urgent need to progress toward intelligent network management systems. Of particular significance in this context is the emergence of network softwarization. By allowing innovative solutions through software-defined networking (SDN) and network function virtualization (NFV), network softwarization represents a radical transformation in the way network services are provisioned and managed. It is this paradigm shift that opens the floodgates for autonomic and intelligent networking, a concept that is rapidly gaining traction within the research community [14], [150].

Intelligent networking, propelled by network softwarization, offers a promising trajectory for network management in the digital era. The softwarization of networks effectively
shatters traditional barriers, thereby facilitating improved control, scalability, and agility. Moreover, it introduces the prospect of autonomic networking - networks that are capable of self-management, thus significantly reducing human intervention and potential errors.

Such advancements are not only pivotal in enhancing network management but are also likely to drive new breakthroughs in the telecommunications industry. This transformation holds the potential to revolutionize the industry, and consequently, it is a matter of utmost importance that researchers, practitioners, and policy-makers alike pay heed to these developments.

The advent of 5G, B5G, and emergent 6G networks has unequivocally established an urgent necessity for significantly reduced latency, robust reliability, and substantial support for a diverse range of devices within the telecommunications domain. Furthermore, the rich heterogeneity of data spawned from 6G networks calls for the deployment of sophisticated mathematical tools capable of extracting pertinent information from the raw data to drive critical decisions. Key among these are resource management and access control decisions, areas where conventional network optimization strategies have been found wanting [49], [127].

Consequently, there is an undeniable imperative for the creation of intelligent network management solutions. By integrating cutting-edge techniques from the domains of machine learning, artificial intelligence, and data analytics, these solutions stand poised to meet the ever-evolving demands of 6G networks and guarantee superior performance. Such intelligent systems could imbue networks with the ability to self-heal, self-configure, self-manage, and self-protect, thereby empowering network operators to administer and manage their networks in a markedly astute and autonomous manner. The end result of this is the provision of a seamless and optimal user experience.

In the pursuit of these ambitious objectives, industry mavens have gravitated towards groundbreaking technologies such as Open Radio Access Networks (O-RAN), Network Function Virtualization (NFV), and Software Defined Networks (SDNs). The O-RAN
alliance, erected upon the cornerstone principles of openness and intelligence, strives to actualize intelligent radio control for the upcoming B5G and 6G wireless networks. Through the application of Artificial Intelligence (AI), it is possible to automate an array of network functions, a move that is anticipated to decrease operating expenses while simultaneously delivering reduced latency, robust reliability, and broad support for an array of devices [115], [137].

However, despite these promising developments, the task of managing complex and highly heterogeneous networks continues to present formidable challenges. The complexity of these tasks underscores the need for continued research and development efforts to explore and implement novel solutions.

Multi-Agent Systems (MAS) have emerged as a potent solution to the inherent complexities in network management, offering a promising route towards intelligent network administration. By harnessing the collective intelligence of multiple agents, MAS hold the potential to optimize network performance, while also significantly enhancing the user experience. Specifically, the adaptive, scalable, and fault-tolerant attributes of MAS are facilitated by the cooperative work of multiple agents in decision-making [158], [48], [12]. Each agent, designed for specific tasks such as monitoring network traffic, detecting and mitigating security threats, or optimizing network performance, continually learns from its experiences and interactions with the network, thereby refining its decision-making abilities over time.

Further, MAS can serve as catalysts for network virtualization, permitting the creation of multiple virtual networks that cater to specific applications or user groups, ultimately curbing network management complexity [162], [14]. This facilitates a more effortless allocation of resources by network managers and ensures that each virtual network adheres to its performance requirements.

Collectively, MAS stands on the brink of a revolution in the telecommunications industry, offering an intelligent, scalable, adaptable, and fault-tolerant approach to network
management. The integration of intelligent management techniques into MAS can significantly bolster the network’s responsiveness, flexibility, and scalability, ensuring optimal performance alongside a seamless user experience. State-of-the-art technologies such as machine learning, artificial intelligence, and data analytics empower network operators to adeptly respond to shifting network conditions, predict future network demands, and proactively address them. Consequently, the embrace of MAS offers a persuasive pathway for network operators to future-proof their networks, enhancing their long-term effectiveness and robustness. Despite these promising prospects, further research endeavors are necessary to fully unearth the immense potential of MAS for intelligent network management.

1.2 Research objectives and contributions

The paramount objective of this research lies in the development and implementation of Multi-Agent Systems (MAS) specifically purposed to enable intelligent network management and engender autonomous network systems. In so doing, the research is projected to provide the much-needed flexibility and agility necessary to satisfy the evolving demands of next-generation networks. The distinct objectives of this research, accordingly, encompass the following key points:

1. To delve into the exploration of the relative merits of an agent-based approach to service design when juxtaposed with conventional micro-service-based designs, thereby unearthing more effective pathways for system implementation.

2. To create a fitting simulation environment capable of efficaciously evaluating the performance of MAS within the context of softwarized networks, providing critical insights into the efficiency and effectiveness of these systems.

3. To introduce a novel, scalable, and comprehensive methodology dubbed the Scalable and Efficient DevOps (SE-DO). This approach aims to optimize the perfor-
mance of intelligent agents, particularly in environments beset with resource constraints, thus enhancing the system’s overall adaptability and responsiveness.

4. To meticulously design and implement a MAS for intelligent network management. The proposed system is expected to infuse intelligence into sub-functions, whilst fostering loosely coupled units in the service-oriented architecture. A particular focus would be cast on the complex realm of radio resource management.

5. To illuminate the promising potential applications of MAS in a tangible manner. Specifically, intelligent traffic management systems could significantly improve network performance, alleviate congestion, and elevate the overall user experience.

6. Finally, the culminating objective of this research endeavors to propose and develop an Autonomous, Intelligent, and advanced AI/ML Framework for Proactive Management and Dynamic Optimal Routing, with the goal of significantly enhancing 6G Network Performance. This novel testbed introduces a Speed Optimized LSTM algorithm. Boasting remarkable speed, this advanced algorithm is designed to predict potential network congestion, thereby enabling preemptive action to maintain optimal network functionality.

Furthermore, including Reinforcement Learning (RL) techniques allows the system to capitalize on these forecasts. This ensures the optimization of routing procedures and the preservation of high-performance levels across the network. As an advanced solution driven by continuous learning and adaptation, this cutting-edge framework aptly mirrors the evolving nature of 6G networks. In this regard, it meets the stringent requirements for ultra-low latency, ultra-reliability, and comprehensive heterogeneity management, which are vital hallmarks of this advanced generation of networks. This final objective thus reflects the research’s commitment to advancing intelligent network management practices in tandem with the ongoing evolution of telecommunications technology.
The completion of these objectives will not only contribute to the academic dis-
course surrounding intelligent network management but also provide tangible solutions
and frameworks that will steer the evolution of network management practices in this
digital era.

1.3 Dissertation outline

This dissertation comprises seven primary chapters, each addressing the research objec-
tives delineated earlier. A structured overview is provided in Table 1.1. The subsequent
breakdown delineates the contents of each chapter:

Chapter I: Introduction

- This chapter provides an overview of the current state of network management and
  control in the telecommunications industry.
- It discusses the challenges and limitations of current approaches in managing com-
  plex and heterogeneous networks.
- The importance of intelligent network management for maintaining network agility
  and flexibility in the digital era is highlighted.
- The chapter emphasizes the emergence of 5G, beyond-5G (B5G), and 6G networks,
  and the requirements they impose on network management and control.
- Finally, the chapter discusses the potential of multi-agent systems (MAS) in ad-
  dressing the challenges of intelligent network management.

Chapter II: Transforming Network Management with Intelligent Approaches

- This chapter presents a comprehensive review of related literature in the field of
  intelligent network management and control.
Table 1.1. Dissertation Structure
• It discusses the state-of-the-art in multi-agent systems, service-oriented architecture, and softwarized networks.

• The chapter also reviews recent developments in machine learning and deep learning algorithms for intelligent network management.

Chapter III: Softwarized Intelligent Network Architecture Design

• This chapter investigates the advantages of an agent-based approach to service design compared to traditional micro-service-based design.

• It proposes a suitable simulation environment for evaluating multi-agent system performance within softwarized networks.

• The chapter concludes with a discussion of the ongoing development and implementation of various network service agents utilizing the Python Agent Development Environment (PADE) framework.

Chapter IV: Scalable and Efficient DevOps for Intelligent Network Management

This chapter proposes a new scalable and comprehensive approach, Scalable and Efficient DevOps (SE-DO), to optimize the performance of intelligent agents in resource-constrained environments. The proposed approach leverages the multi-agent-based service design, providing both reactive response and proactive anticipation and reconfiguration of the network system to suit the dynamic requirements of the network.

Chapter V: Quality of Service (QoS) and Radio Resource Management (RRM)

• This chapter introduces QoS and RRM tasks and highlights their importance in network management.

• It presents the design of intelligent QoS agents for capturing and responding proactively to network traffic and workload distribution.
The chapter proposes a ranking mechanism for allocating the best path proactively for QoS, and introduces a new scheduling algorithm for balancing throughput, fairness, and user QoS in the multi-channel case using multi-agent reinforcement learning (MA-RL).


- This chapter embarks on a comprehensive exploration of leveraging Artificial Intelligence and Machine Learning (AI/ML) for autonomous and intelligent 6G network management, highlighting the significant role of Speed-optimized LSTM (SP-LSTM) and Reinforcement Learning models.

- It outlines the transformative potential of AI/ML-based agents for proactive management and dynamic routing, emphasizing the necessity for advanced systems in dealing with complex, latency-sensitive 6G networks.

- The chapter delves into an in-depth analysis of the novel SP-LSTM architecture, revealing its superiority over conventional LSTM models in predicting network congestion and managing network performance.

- A discourse on the role of Reinforcement Learning within autonomous network management systems and its capability to handle vast state spaces and high-dimensional data is presented.

- It provides a detailed exposition of the amalgamation of SP-LSTM-based predictive analytics and RL-based dynamic routing for robust and adaptive network management, leading to improved network resilience and performance.

- The chapter elucidates the influential role of multi-agent systems (MAS) in 6G network management, underscoring the potential of integrating MAS with reinforcement learning for efficient and context-aware routing decisions.
• It presents empirical validation of the proposed AI/ML architecture for network management through a comprehensive use case, demonstrating superior results in terms of prediction accuracy and computational efficiency.

• The chapter concludes by suggesting future work directions, including multi-agent reinforcement learning (MARL) and edge computing integration into the multi-agent system framework, setting the stage for robust, adaptable network systems in the 6G era and beyond.

Chapter VII: Conclusion and Future Work

• This chapter summarizes the dissertation’s contributions, limitations, and achievements.

• It provides recommendations for future research directions and emphasizes the potential of the proposed approach in meeting the challenges of next-generation networks.
Chapter 2

Transforming Network Management with Intelligent Approaches

This compelling chapter introduces an innovative approach to network management, utilizing advanced technologies such as AI, ML, and DL to address increasing complexity in the field. The work underscores the implementation of intelligent agents into a microservices-based SDN controller, including new functionalities such as topology learning and shortest path calculation. The experimentation and practical implementation, carried out in conjunction with a Master of Science student that I supervised, validate the substantial potential and efficacy of these intelligent network management approaches.

2.1 Introduction

The construct of intelligent network systems has been the subject of rigorous academic discourse and multifarious investigation within the literature [74]; [53]; [93]; [73]; [39]; [136]. The striking surge in the integration of interconnected devices, sensors, and appliances has precipitated an escalating complexity in network management that has seen traditional methods strained to their limits [37]; [29]. The continuous surge in the volume and diversity of data conveyed across these networks consequently amplifies the task of effective network control and management.

Nonetheless, the innovative strides made in the realm of intelligent network management offer propitious solutions to these burgeoning challenges [4]; [62]. Intelligent network management incorporates the utilization of avant-garde technologies, such as Artificial Intelligence (AI), Machine Learning (ML), and Deep Learning (DL). These technologies stimulate autonomous decision-making processes and catalyze network performance optimization [97]; [165]. Such technological innovation empowers network managers by
providing them with real-time, insightful observations into network performance. Consequently, network managers are better equipped to make data-driven, informed decisions regarding network control and management [14]; [60].

This chapter unfolds with a comprehensive review of extant literature in the domain of intelligent network management and control. This review critically appraises the state-of-the-art developments in multi-agent systems, service-oriented architecture, and Software-Defined Networks (SDN), contributing to a holistic understanding of current advancements and future directions in the field.

2.2 Softwarized network architecture

Service-oriented architecture (SOA) and software-defined networks (SDN) are promising approaches to intelligent network management. SOA enables the creation of loosely coupled services that can be combined to provide flexible and scalable network services. SOA can facilitate the integration of different network technologies, making it easier to manage complex and heterogeneous networks. Additionally, SOA can provide better support for dynamic network provisioning and resource allocation, allowing for more efficient use of network resources. Softwarized network architecture, including SOA, Network Function Virtualisation (NFV), and SDN, can work together in achieving dynamic network function chaining through various tasks such as service composition specification, service selection, delivery, and placement. In an edge/cloud-based distributed environment, the core components for network softwarization, decomposition, and orchestration are SDN and NVF [54], [106]. Figure 2.1 illustrates the interconnection between SOA, NFV, and SDN, which enables network service providers (NSPs) to offer more effective service delivery models, while Table 2.1 provides a structured comparison across different architectural paradigms used in networking and software development, namely Service-Oriented Architecture (SOA), Network Functions Virtualization (NFV), and Software-Defined Networking (SDN). This helps NSPs remain competitive and keep pace with the
service offers and infrastructure innovations of over-the-top (OTT) providers [101].

Software-Defined Networking (SDN) is a promising approach to intelligent network management and control. It provides a centralized approach to network control and management, which enhances network agility and reduces network management complexity. With SDN, as illustrated in Figure 2.2, network managers can decouple the control and data planes, which enables them to more easily add, remove, or modify network functions and policies. This allows for more efficient resource allocation and utilization, which is crucial in the dynamic and ever-changing world of telecommunications.
Table 2.1. Comparison of Different Networking and Software Architectures

<table>
<thead>
<tr>
<th>Concept</th>
<th>Scope</th>
<th>Primary Use</th>
<th>Components</th>
<th>Management</th>
<th>Characteristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOA</td>
<td>Enterprise-level, spans across different departments and applications.</td>
<td>To integrate disparate systems and enable them to communicate with one another.</td>
<td>Services that are loosely coupled and communicate via standard protocols like SOAP.</td>
<td>Centralized service directories.</td>
<td>High-level of abstraction, can be stateful or stateless, often synchronous communication.</td>
</tr>
<tr>
<td>NFV</td>
<td>Network services in telecommunication and data centers.</td>
<td>To decouple network functions from hardware, making it possible to deploy and manage network services using software.</td>
<td>Virtual Network Functions (VNFs) that replace traditional hardware-based network functions.</td>
<td>Manages the lifecycle of network functions.</td>
<td>Designed to be modular and scalable but focused on networking.</td>
</tr>
<tr>
<td>SDN</td>
<td>Network infrastructure primarily in data centers.</td>
<td>To separate the control plane from the data plane in networking devices.</td>
<td>SDN Controller, SDN Applications, and Network Devices.</td>
<td>Centralized management via SDN Controller.</td>
<td>Focuses on programmability and centralized network provisioning.</td>
</tr>
</tbody>
</table>
Figure 2.2: The architecture of Software-Defined Networking (SDN): enables a centralized and programmable approach to network control and management. By separating the control and data planes, SDN provides network managers with greater flexibility and agility to add, remove, or modify network functions and policies, resulting in more efficient resource allocation and utilization. Adapted from [116].

One of the key benefits of SDN is its ability to support network virtualization, which enables the creation of multiple virtual networks that can be tailored to specific applications or user groups. This approach can significantly reduce network management complexity by allowing the creation of more focused networks. By using SDN for network virtualization, network managers can more easily allocate resources and ensure that each virtual network meets its performance requirements.

Moreover, the use of SDN can facilitate the implementation of intelligent network management and control through the use of machine learning and deep learning algorithms. The advent of SDN has opened up novel prospects for integrating intelligence within networks [1]. For example, Kumar et al. [80] proposed a machine learning-based traffic engineering approach that leverages SDN to improve network resource utilization and reduce network congestion. This approach uses a reinforcement learning algorithm to dynamically adjust network traffic routing based on current network conditions, re-
resulting in more efficient use of network resources.

Another benefit of SDN is its ability to support network programmability and automation. With SDN, network managers can use programming languages and software tools to automate network management tasks, reducing the need for manual intervention and improving network reliability and efficiency. For example, the Open Networking Foundation’s ONOS project [24] is an open-source SDN controller that provides a platform for developing and deploying network applications and services.

In brief, SDN is a powerful approach to intelligent network management and control, providing centralized control and management that enhances network agility and reduces network management complexity. It enables network virtualization, machine learning-based traffic engineering, network programmability, and automation, which are crucial for efficient resource allocation and utilization in today’s telecommunications industry.

2.3 Charting next-generation networks: The Rise of Softwarization and In-Network Intelligence

The ushering in of 5G technology has significantly accelerated the transformation of network architecture into software-defined platforms. This represents a marked departure from the classical data-storage-and-transfer model towards an advanced computational and transference paradigm. As such, computation plays an increasingly vital role alongside communication in determining the trajectory of future networking technologies [55].

Moreover, the International Telecommunication Union (ITU) has recently released a forward-looking report on technological trends poised to shape the years up to 2030 and beyond [71]. A salient theme of this report is an overview of the forthcoming 6G networks. The European Union, in 2021, undertook significant strides in 6G research and development, as evinced by its flagship project Hexa-X and the German 6G-life research hub. Notably, the Hexa-X project continues to make significant contributions, laying down the fundamental principles concerning the features, applications, key performance
benchmarks, and design of 6G [50], [52].

Current endeavors, however, are centered around network softwarization and in-network intelligence. Such initiatives have been fruitful, shedding light on and providing valuable input toward the structural facets of future network generations [16].

2.4 Revolutionizing SDN Controllers: Microservices & Multi-Agent Systems

2.4.1 Microservices-based SDN

The fundamental concept of the microservices-based SDN (MSN) framework centers on achieving network information and state synchronization, leading to a comprehensive understanding of the network. This strategy fosters independent deployment and encourages the reuse of components. The MSN methodology necessitates the deconstruction of an SDN controller, as depicted in Figure 2.3. The diagram displays a tri-layered disaggregated SDN architecture, mirroring an NFV structure. The control layer is compartmentalized into several subfunctions, all of which are actualized as software-driven network functions. Renowned orchestrators such as ETSI MANO or Kubernetes can streamline these subfunctions, thereby facilitating a service function chain that effectively mirrors the operations of the conventional SDN controller. As seen in Fig. 2.3, the upper stratum consists of an assortment of autonomously implemented SDN modules. Each of these modules serves a unique purpose, such as topology management or routing. These operations can be divided into basic SDN controller functions and supplementary functions or applications. Basic SDN controller functions are indispensable for mimicking the least possible functionality of an SDN controller. Supplementary functions can be viewed as applications including firewall and QoS monitoring.
Figure 2.3: MSN Decomposition Architecture.

2.4.2 Multi-Agent-based SDN

Multi-agent systems (MAS) have emerged as a highly promising solution to the challenges of intelligent network management, offering numerous advantages over traditional network management approaches. MAS allows multiple agents to work together to make decisions, providing better adaptability, scalability, and fault tolerance [10]. These agents can be designed to perform specific tasks, such as monitoring network traffic, detecting and mitigating security threats, or optimizing network performance, enabling network managers to achieve better control over network operations.

One significant advantage of MAS is their ability to learn from their experiences and interactions with the network. This ability allows agents to improve their decision-making
abilities over time, making them more effective at managing complex and heterogeneous networks. Additionally, MAS provides better fault tolerance, as agents can continue to perform their designated tasks even if other agents fail or are unavailable, reducing the risk of network downtime.

MAS is also highly beneficial for network virtualization, which enables the creation of multiple virtual networks that can be tailored to specific applications or user groups. By using MAS for network virtualization, network managers can more easily allocate resources and ensure that each virtual network meets its performance requirements. This approach significantly reduces network management complexity, enabling network managers to focus their efforts on optimizing network performance and enhancing user experiences.

Moreover, recent advances such as deep reinforcement learning (DRL) have shown significant potential for intelligent network management using MAS. DRL combines the advantages of reinforcement learning with deep neural networks, allowing agents to learn from their experiences and interactions with the network in a more efficient and effective manner. This approach enables agents to make more informed decisions, leading to improved network performance and a better user experience. Li et al. (2022) suggest that using multi-agent reinforcement learning for intelligent networking leverages the collective intelligence of multiple agents to optimize network performance and provide better user experiences [86].

Additionally, MAS enables the deployment of distributed intelligent systems, which can operate in real-time to provide better network management and control. For example, agents can be designed to perform real-time network analysis and provide insights into network performance and potential security threats. MAS can enable the creation of a collaborative and adaptive network management system that can dynamically adjust to changing network conditions [153].

Despite the scarcity of initiatives utilizing multi-agent systems in the context of SDN
Controllers, there has been a recent development and employment of a framework designed to evaluate the effectiveness of multi-agent systems within such software-centric networks [151]. A project termed MASDN (MultiAgent SDN) has recently introduced an SDN Controller architecture that relies on multi-agent systems [35]. This MASDN controller deviates from the traditional, monolithic structure and is composed of multiple discrete agents. These agents, though independent, communicate amongst themselves via inter-process communication (IPC) or TCP. The controller’s nucleus consists of an agent that forms a part of the multi-agent system and communicates with agents responsible for running various network applications. As a result, the applications are freed from being bound to a specific Southbound Interface (SBI). The agent system can either be centrally deployed on a single physical machine or distributed across several physical and virtual devices (for instance, containers), making use of a shared knowledge base for the preservation of coherent information. Nonetheless, there currently exists no implementation to examine and enhance the proposed architectural design [35].

![Figure 2.4: An Overview of the microservice-based SDN Architecture with Intelligent Agents](image)

Figure 2.4: An Overview of the microservice-based SDN Architecture with Intelligent Agents
Even though microservices play a pivotal role in large-scale software development, they exhibit striking similarities to agents [155]. Notably, future network architectures utilize Multi-Agent Systems (MAS) to achieve full autonomy [14]. Furthermore, the concept of network function atomization revolves around identifying the smallest viable units of network functions within a system predicated on microservices. Despite their resemblance to microservices, MAS boasts a higher level of independence and the ability to proactively act. Figure 2.4 showcases a comprehensive diagram that demonstrates the incorporation of intelligent agents within the Microservice-Based SDN Architecture.

2.4.3 Smart Topology Learning in MSN Framework: Unleashing Intelligent Agents

This section elucidates the implementation intricacies involved in understanding topology learning within the MSN paradigm. Emphasis is placed on the microservice aspect of network topology knowledge acquisition, designed to collect and assimilate data associated with network structure conveyed by the ofp emitter, which primarily identifies events labeled as EventSwitchEnter. The process then involves the formation of a graph, symbolizing the switches along with the associated links, and host-representing nodes. In tandem, the topology learning knowledge acquisition requires an update in the awareness of the shortest-path agent, facilitated by interaction with a basic switch. The MSN framework’s capacity to manage intelligent agents for topology management support is depicted in Figure 2.5. We employ a graph-based approach to encode the information, utilizing the NetworkX library that provides features for graph enhancement. To facilitate the learning of network structures, thereby ensuring updated information for the shortest path algorithm, a unique REST feature designated as /updategraph has been established. This feature modifies the graph via a straightforward switch REST microservice. A depiction of the update procedure is presented in Figure 2.5. Simultaneously, the shortest path agent is designed to ascertain the briefest route using a particular method-
Figure 2.5: Updating shortest path agent via topology learning service

Our system applies both the Dijkstra and the Bellman-Ford algorithms for this task. Specifically, when a new /packetin event is triggered by the ofp emitter and transmitted to the simple switch rest, the shortest path agent employs its knowledge base to identify the shortest path. This action triggers an event to update the rule set, which is forwarded to the controller for additional node communication, particularly involving the /stats/flowentry/add REST event. In instances where the shortest path agent lacks information about the necessary switch port for packet forwarding, it defaults to using the simple switch rest OFPP FLOOD as the output port. This port is a special ALL port within OpenFlow switches, permitting packet forwarding to all ports with the exception of the input port.

Moreover, this specific version of the MSN implementation, which includes the topology learning microservice and the shortest path agent extension, is publicly accessible. It can be found by the community at the following URL: https://gitlab.com/dscotece/ryusdnndecomposition/-/tree/agents?reftype=heads.

The research infrastructure used for this study is a virtual Linux machine (Ubuntu

Furthermore, we utilized a Docker-oriented variant of the MSN platform [134]. To gauge the performance of the microservice responsible for network topology understanding, we investigated five unique network structure designs, as depicted in Figure 2.6:

- A singular-tier hierarchical tree, incorporating 1 switch and 2 hosts;
- A dual-tier hierarchical tree, integrating 3 switches and 4 hosts;
- A tri-tier hierarchical tree, comprising 7 switches and 10 hosts;
- A mixed network, containing 6 switches and 7 hosts;
- A star network, housing 1 switch and 5 hosts.

The varying network configurations were achieved via the Mininet simulator utilizing Python APIs. The experimental results recorded are averages from 30 trials, displaying a nominal variability of less than 5%.

In our inaugural experiment, we focused on assessing the temporal effects introduced by the topology learning microservice. Specifically, we examined the total time required to enable communication between various hosts within a three-tier hierarchical tree network configuration. To facilitate this, we employed a straightforward client-server script, with the server script operational on the H1 node, and the client script running on H7 and H3 hosts. As illustrated in Figure 2.7, the aggregate time for the initial packet encompasses the duration necessary for pinpointing the hosts’ location from the topology learning, followed by the time required to establish the rule in the controller. Once this stage is reached and until the rules embedded within the switches expire, all packets sent between H1 and H7 (or H1 and H3), and the reverse, can be delivered by the network framework without additional controller engagement. We term these packets as the
"Normal flow." Ultimately, when the rules reach their expiration, the SDN Controller is tasked with renewing the flow rules' validity, a process consuming time similar to the initial packet delay.

Subsequently, we assessed the cumulative time required to generate the knowledge within the topology learning microservice in relation to various network topologies. As previously mentioned, the knowledge derived from topology learning takes the shape of a graph. As shown in Figure 2.8, the documented timings encapsulate the gathering of switches and hosts as well as the formation of the graph's nodes and links. It's clear that the scale and intricacy of the topology significantly impact the duration required. For example, in the context of the star and 1-layer hierarchical tree topologies, the time frame related to the aggregation of switches (both have one) is comparable. However, when it comes to gathering hosts, the star topology, which possesses more hosts, demands a
Figure 2.7: Communication delay between a pair of nodes in a client-server setup

Figure 2.8: Discovery Time for Different Network Topologies

lengthier duration.

In contrast, we conducted a comprehensive analysis of the shortest path agent’s con-
duct. As addressed in the preceding section, the graph illustrating the topology serves as
the foundational knowledge for potentially creating a model that intelligently determines
the shortest path within the agent. Specifically, Figure 2.9a displays the cumulative size of the knowledge in bytes, encompassing all the updates. The subsequent graph, shown in Figure 2.9b, signifies the total count of updates received by the shortest path agent, including repeated identical updates resulting from recurrent events (each switch triggers the event, but from the first or second event onwards, all data can be retrieved). Finally, Figure 2.9c provides a breakdown of the overall duration of all updates, differentiated by the network topology employed. It’s crucial to highlight that the complexity of the network topology invariably influences the behavior of the shortest path agent.

2.5 Intelligent Network Management in Next-Generation Networks

As the world advances towards digitization, the rapid increase in interconnected devices and the consequent exponential rise in data traffic present considerable challenges to network management. Traditional network management paradigms often struggle with the massive complexity and dynamic nature of these networks. Intelligent network management, backed by machine learning (ML), deep learning (DL), and reinforcement learning (RL) algorithms, is becoming an effective response to these challenges. With 6G networks on the horizon, it is pivotal to investigate and fully understand how ML, DL, and RL algorithms can support intelligent network management, optimizing network performance, and user experience [22].
2.5.1 Machine Learning for Network Management

ML techniques form the backbone of many modern intelligent network management systems, owing to their ability to efficiently analyze large volumes of data and predict network behavior. Supervised learning, a common ML approach, uses labelled data to predict outcomes or classify data points. Algorithms such as linear regression, decision trees, random forest, and support vector machines are frequently used to predict network traffic and optimize network configuration settings [17]. By efficiently analyzing traffic data, these ML algorithms can help network managers anticipate network demands, allocate resources optimally, and adjust network configurations, thus improving network throughput and reducing congestion.

In contrast, unsupervised learning operates on unlabeled data to discern inherent patterns or anomalies. Clustering techniques such as K-means and hierarchical clustering and dimensionality reduction techniques like Principal Component Analysis (PCA) are popular unsupervised learning methods used in network management [13]. They can identify unobservable patterns in network traffic data, contributing to the enhancement of network security and performance by detecting anomalies and potential security breaches.

Furthermore, semi-supervised learning algorithms, which leverage both labeled and unlabeled data, provide a viable approach for network traffic classification, anomaly detection, and resource allocation [8], [168]. In scenarios where obtaining labeled data is scarce or expensive, semi-supervised learning algorithms offer a means to augment the training process, leading to performance improvements.

2.5.2 Deep Learning for Anomaly Detection and Security Threat Mitigation

Deep Learning, a subfield of machine learning that uses artificial neural networks with multiple layers (i.e., "deep" networks), has shown its prowess in tasks such as anomaly detection and security threat mitigation. Deep Neural Networks (DNNs), Convolutional
Neural Networks (CNNs), and Recurrent Neural Networks (RNNs) are capable of modeling complex, non-linear relationships in high-dimensional data, making them ideal for processing large-scale network data [58].

CNNs have been successful in identifying network intrusions by analyzing raw network traffic data [89]. RNNs and their variants such as Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) can process temporal sequences, making them suitable for detecting anomalous behavior in time-series network data [68]. These algorithms enable network managers to identify abnormal network behavior accurately, offering proactive security measures against potential threats.

2.5.3 Reinforcement Learning for Real-Time Network Decisions

Reinforcement Learning, a type of machine learning where an agent learns to make decisions by interacting with its environment, is increasingly recognized for its potential in intelligent network management, particularly in 5G and future 6G networks. RL optimizes the performance of an agent based on the feedback (reward or punishment) it receives from the environment, enabling it to make better decisions over time [144].

RL can support real-time network resource allocation, congestion control, and user experience optimization. For instance, RL can be used to optimize key performance indicators (KPIs) across multiple domains, such as the Radio Access Network (RAN), Core, and Orchestration/Operations, Administration, and Maintenance (OAM) [111].

With its adaptive and flexible nature, RL can reduce the overall network management complexity, enable efficient resource allocation, and enhance user experiences. It can also predict network traffic patterns and proactively adjust network parameters to prevent congestion and ensure the quality of service (QoS).

Moreover, RL can help automate various network orchestration and management tasks, including network slicing, virtual network function placement, and service function chaining. By minimizing human intervention, RL provides a more efficient and automated
approach to network management, enabling more agile network operations.

Multi-agent reinforcement learning (MARL), an extension of RL, can leverage the collective intelligence of multiple agents to optimize network performance and provide better user experiences [6]. It involves training multiple agents to interact with each other and the network environment, learning from their experiences to optimize network performance. This approach has shown promise in various network management tasks, including network resource allocation, traffic routing, and load balancing.

2.5.4 Hybrid and Ensemble Learning Approaches

While individual ML, DL, and RL techniques provide effective solutions for various network management tasks, their combination, often referred to as hybrid models, can harness the strengths of each approach to tackle complex networking challenges. For instance, a hybrid of RL and supervised learning can manage network resources by leveraging supervised learning for prediction tasks and RL for decision-making tasks [94].

Ensemble learning techniques, such as boosting and bagging, also hold considerable promise for network management. By aggregating the decisions of multiple models, ensemble methods can achieve superior prediction accuracy and robustness, improving network traffic prediction, and security threat mitigation [7], [96].

2.5.5 Federated Learning for Decentralized Learning

With 6G networks’ emphasis on decentralization and edge computing, federated learning, a distributed machine learning approach, has attracted attention. In federated learning, a global model is trained across multiple devices or servers holding local data samples, obviating the need to centralize data and thus enhancing user privacy and reducing communication costs [31], [166]. As the complexity and scale of future networks continue to grow, federated learning could become a crucial tool for intelligent network management in 6G networks.
The pivotal concepts delineated in this section are concisely encapsulated in Table 2.2 for quick reference and better comprehension.

Table 2.2. Overview of ML, DL, and RL Techniques and Applications in Network Management

<table>
<thead>
<tr>
<th>Main Area</th>
<th>Techniques</th>
<th>Key Concepts and Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Machine Learning for Network Management</td>
<td>Supervised learning (e.g., SVM)</td>
<td>Supervised learning for traffic prediction and configuration optimization</td>
</tr>
<tr>
<td></td>
<td>Unsupervised learning (e.g., PCA)</td>
<td>Unsupervised learning for identifying patterns and security breaches</td>
</tr>
<tr>
<td></td>
<td>Semi-supervised learning</td>
<td>Semi-supervised learning for traffic classification, anomaly detection, resource allocation</td>
</tr>
<tr>
<td>Deep Learning for Anomaly Detection and Security Threat Mitigation</td>
<td>DNNs, CNNs, RNNs</td>
<td>CNNs for network intrusion identification</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RNNs for anomalous behavior detection in time-series data</td>
</tr>
<tr>
<td>Reinforcement Learning for Real-Time Network Decisions</td>
<td>Standard Reinforcement Learning</td>
<td>RL for real-time network resource allocation, congestion control, user experience optimization</td>
</tr>
<tr>
<td></td>
<td>Multi-agent Reinforcement Learning (MARL)</td>
<td>MARL for collective intelligence in network performance optimization</td>
</tr>
<tr>
<td>Hybrid and Ensemble Learning Approaches</td>
<td>Hybrid Models Ensemble Learning (Boosting, Bagging)</td>
<td>Hybrid models for complex tasks</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Ensemble methods for improved prediction accuracy and robustness</td>
</tr>
<tr>
<td>Federated Learning for Decentralized Learning</td>
<td>Federated Learning</td>
<td>Distributed machine learning enhancing user privacy and reducing communication costs</td>
</tr>
</tbody>
</table>

2.6 Chapter Summary

This chapter has embarked on an insightful journey into the complex world of intelligent network management, unraveling the rich tapestry of technologies, approaches, and paradigms that form its foundation. Our exploration centered around the integration of intelligent agents within a microservices-based Software-Defined Networking (SDN) con-
troller, a proposition that promises to revolutionize our understanding and management of ever-evolving network ecosystems.

We commenced with an expansive literature survey, probing into the accelerating trend of device and sensor integration, a phenomenon that is intricately weaving a dense network tapestry, rendering its management increasingly complex. Through our discussions, we underscored the potent solutions offered by intelligent network management, harnessing avant-garde technologies like Artificial Intelligence (AI), Machine Learning (ML), and Deep Learning (DL) to enable autonomous decision-making processes and optimize network performance.

An exploration of the Service-Oriented Architecture (SOA) and SDN in the realm of intelligent network management painted a vivid picture of a dynamically adaptable and scalable network ecosystem. The softwarization of network architecture, manifested in SOA, Network Functions Virtualization (NFV), and SDN, is pioneering a shift towards dynamic network function chaining, thereby enhancing service delivery models and heightening competitiveness.

Within this transformative landscape, we magnified the role of SDN, dissecting its potential in augmenting network agility, simplifying complexities, and facilitating network virtualization. The incorporation of ML and DL within the SDN paradigm was identified as a crucial catalyst in improving network resource utilization and mitigating congestion, leading to a more efficient and reliable network structure.

Furthermore, the spotlight was cast on the potential of microservices-based SDN (MSN) and multi-agent systems (MAS) to revolutionize SDN controllers. We delved into the MSN approach, an architectural style that encourages network state synchronization and promotes component reuse, thereby providing a more comprehensive understanding of the network state. Simultaneously, the power of MAS was illuminated, outlining its potential in bolstering decision-making, network adaptability, scalability, and fault tolerance, offering significant advantages in network virtualization.
Advancing further, the application of deep reinforcement learning (DRL) within the MAS framework was elucidated, illuminating its capacity to enhance network performance and user experience. Despite the under-exploitation of MAS in SDN controllers, we underscored the encouraging strides in this direction, exemplified by recent advancements such as MASDN.

In the final analysis, this chapter underlined the transformative role of intelligent technologies in network management, particularly within the SDN context. It provided a blueprint for future explorations, elucidating the critical function of these technologies in preparing the ground for future practical implementations. The pivotal roles of microservices, MAS, and intelligent agents, with their inherent capacity for autonomous and proactive action, were recognized as game-changers in the design of future network architectures.
Chapter 3

Softwarized Intelligent Network Architecture Design

This chapter elucidates the design and optimization of a Softwarized Intelligent Network Architecture, underpinned by multi-agent systems (MAS). With the growing demand for highly reliable, efficient, and resilient network systems, our design offers an innovative solution by incorporating intelligence into the network architecture. The MAS-based platform is intended to augment current network systems, allowing them to adapt, anticipate, and swiftly address potential issues while optimizing performance. This chapter seeks to highlight the potential of MAS-based intelligent network platforms to revolutionize current network systems by providing a robust, adaptive, and highly optimized solution. We hope it paves the way for further exploration and development in this exciting and important area.

3.1 Monolithic and Microservice Architectures: A Comparative Study of Benefits and Challenges

The detailed depiction in Figure 3.1 succinctly contrasts the salient features of monolithic and microservice architectural styles. Historically, monolithic architecture was the preferred architectural modality until recent breakthroughs such as cloud services and Kubernetes started gaining traction [113].

Monolithic architecture signifies a cohesive, indivisible software application, traditionally disseminated via physical mediums such as CD-ROM, and typically updated annually. This architectural style is often criticized for its inertia in accommodating changes, high operational overhead, and lack of adaptability towards distinct or evolving product requirements [84].
Even minute modifications within a monolithic codebase could potentially trigger the need for a comprehensive overhaul and deployment of the entire software, engendering a bloated development cycle [114]. Furthermore, scaling selective functionalities in a monolithic application involves an expansion of the entire application, thus complicating the process of updates and incremental scaling efforts [128].

To circumnavigate these issues, microservices emerged as a modern, transformative architectural style. Microservices offer a solution to the drawbacks of monolithic systems, promoting modularity, facile updates, and targeted scaling [109]. By disintegrating a monolith into manageable, independent services, microservices foster adaptability and ease of maintenance, thereby empowering organizations to better align with evolving business needs [119].

![Figure 3.1: Contrasting Monolithic and Microservice Architectures](image)

Figure 3.1: Contrasting Monolithic and Microservice Architectures: Monolithic architecture is a traditional approach where an entire application is distributed as a single unit, making changes slow and costly. In contrast, microservices architecture divides an application into smaller, independently deployable services, facilitating quicker and simpler modifications.
3.1.1 Unearthing the Superiority of Multi-Agent Systems over Microservices: An In-depth Examination

The paradigm of microservices employs the strategic disintegration of large-scale applications into discrete, self-sufficient components, or microservices, deployed over a network. This innovative approach, favored in cloud-based applications, empowers the creation of nimble, scalable systems, promotes upgradability, amplifies fault tolerance, and simplifies testing procedures. Nevertheless, it does not come without its challenges, encompassing facets such as decomposition, orchestration, communication, and integration testing overheads [36], [112].

In recent years, a burgeoning interest in the agent-based approach to service design has emerged as a compelling alternative to the prevalent microservice-oriented design [155]. The agent-based model pivots on the foundation of autonomous agents interacting amongst themselves to execute tasks and realize objectives, offering a host of advantages over conventional microservice designs.

Foremost, agent-based strategies are inherently equipped to support dynamic and unpredictable environments more effectively. While traditional microservices are relatively static with predefined APIs invoked by other services, agents showcase dynamism and adaptability, autonomously interacting with their peers in a sophisticated, intelligent fashion to accommodate environmental flux.

Additionally, agent-based methodologies cater more efficiently to the demands of complex and distributed systems. The increasing intricacy of contemporary systems often gives rise to challenges such as service dependencies, versioning, and coordination quandaries in conventional microservice architectures. The agent-based model mitigates these issues, fostering decentralized and autonomous inter-agent interactions, thereby diminishing the reliance on centralized coordination and control.

Further, the incorporation of intelligent and adaptive systems is more streamlined in agent-based frameworks. Agents, empowered by machine learning and other AI technolo-
gies, are capable of learning and adjusting to shifting conditions, thereby enabling the establishment of intelligent systems capable of responsive decision-making and action-taking in alignment with dynamic environmental conditions and user requirements.

However, the transition to agent-based systems does not come without its hurdles. The design and implementation of autonomous agents often demand increased complexity and resource investment when compared to traditional microservices. Furthermore, security and privacy considerations can be intensified in agent-based models due to the potential of agents accessing sensitive data and systems.

Collectively, the agent-based model presents significant advantages over traditional microservice designs, particularly in dynamic, complex, and intelligent systems. Its growing popularity heralds its potential as a promising approach in the design of future-gen systems. Table 6.4 offers a comprehensive amalgamation of microservices and multi-agent system definitions, thereby elucidating the correlation between the principles of microservices and MAS.

3.1.2 Architecting Intelligent Networks for Next-Generation Connectivity: An Exploration

As delineated in the prior section, the adoption of an agent-based approach to service design presents a compelling advantage over the traditional microservice-oriented design. Not only confined to reactive responses, agent-based services exhibit the ability to preemptively discern and rectify potential issues, paving the way for a drastic reduction in downtime, bolstering system resilience, and enhancing user experience.

Furthermore, the incorporation of multi-agent-based intelligent network service design has emerged as a trending practice, marking an avant-garde shift in network research. This pioneering methodology offers considerable benefits, including scalability, flexibility, and adaptability. Within the multi-agent framework, each agent is accountable for a particular task, and collectively, they synergize to accomplish the system’s overall goals.
Table 3.1. Comparison of Microservices and MAS.

<table>
<thead>
<tr>
<th>Principle</th>
<th>Microservices</th>
<th>MAS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Defining Contextual Boundaries</td>
<td>A microservice is a single business function.</td>
<td>An agent can perform one or more roles in a system.</td>
</tr>
<tr>
<td>Size</td>
<td>Microservices need to be small to guarantee that they are easy to maintain and expand.</td>
<td>MAS research is not concerned with the size and complexity of systems and depends on the specific domain.</td>
</tr>
<tr>
<td>Private state information</td>
<td>Sharing state information is reduced in services.</td>
<td>Agents maintain local and private state, which is essential for their autonomy.</td>
</tr>
<tr>
<td>Spread across multiple nodes</td>
<td>Services are deployed across several nodes.</td>
<td>The distribution of agents is expected to be logical and spread out over multiple nodes.</td>
</tr>
<tr>
<td>Automated control of the system</td>
<td>Automated operations manage failures and scaling.</td>
<td>Management operations are not a primary concern for agents, though they may be taken into account.</td>
</tr>
<tr>
<td>Elasticity</td>
<td>The application can add or remove resources while it is running.</td>
<td>Dynamic adjustment of the number of agents during runtime is an important aspect of MAS.</td>
</tr>
<tr>
<td>Flexible connections</td>
<td>Decomposition of systems into cohesive and loosely coupled services.</td>
<td>Agents operate independently and solve problems with loose coupling.</td>
</tr>
<tr>
<td>Autonomy</td>
<td>Microservices function independently without human intervention and have control over their internal state and actions.</td>
<td>Agents function autonomously without human intervention and control their actions and internal state.</td>
</tr>
<tr>
<td>Social capacity</td>
<td>Microservices usually communicate with each other by exchanging messages through RESTful APIs and HTTP.</td>
<td>Agents communicate with each other by utilizing an Agent Communication Language.</td>
</tr>
<tr>
<td>Reactivity</td>
<td>Microservices are designed to handle incoming requests using HTTP in a timely manner.</td>
<td>Agents sense and timely respond to changes in their environment.</td>
</tr>
<tr>
<td>Proactivity</td>
<td>Microservices are reactive and do not initiate actions.</td>
<td>Agents act autonomously, taking the initiative to interact with their environment.</td>
</tr>
</tbody>
</table>
This strategy fosters heightened autonomy and decentralization, yielding a more robust and efficient system. The inherent intelligence of agents enables them to adapt to shifting environments, thereby enhancing the system’s resilience and self-healing capabilities [133].

Thus, the implementation of a multi-agent-based model in network service design presents substantial benefits over traditional microservice-oriented design. This promising paradigm has the potential to amplify system performance, enrich user experience, and catalyze innovation in network research.

This section introduces an innovative methodology for crafting intelligent network architectures for forthcoming networks such as 6G. In place of conventional hardware-based solutions, we advocate for a software-based approach that employs agents as the foundational elements of the network. By capitalizing on the distinctive characteristics of agent-based systems, such as autonomy and adaptability, we can architect a network framework that boasts significant scalability and exhibits resilience to variations in traffic patterns and user demands [33].

### 3.2 Advanced Internal Architecture of Agents: A Cognitive Leap

This investigation propels the internal architecture of agents to unprecedented heights by integrating sophisticated cognitive components that augment decision-making capabilities in network management [150]. The architecture showcases robust input and output proficiencies, empowering agents to meticulously monitor and analyze the network state while exerting fine-tuned control over the network.

These cognitive components function as the intellectual core of the agents, facilitating strategic and informed decision-making based on the assimilation of knowledge. The decision outcomes are subsequently morphed into actionable tactics by the planning component, whereas the validation component operates as a vital checkpoint to verify the suitability of enacted actions. This stratified approach to network automation engenders
unparalleled precision, efficacy, and reliability in network management, as depicted in Figure 3.2. The figure underscores the agents’ proficiency in administering the network in compliance with predetermined requisites.

Through this groundbreaking architecture, the optimization of network management can be realized, thereby guaranteeing a more efficient and dependable network for users.

Figure 3.2: Cognitive-Driven Agent Architecture: Agent Internal Architecture showing the cognitive, planning, and validation components that enable informed decision-making and precise control over network management. Adapted from [150].

3.3 Elucidating the System Architecture: A Multi-Agent Perspective

Within the purview of our network management system, we leverage various types of agents as integral building blocks to architect a robust and encompassing system. The agents are meticulously designed and synthesized to cater to the precise system pre-
requisites and functionalities. Our architecture, grounded in the multi-agent-based autonomous network management philosophy, illustrated in Figure 3.3, incorporates a broad spectrum of agents, each endowed with distinct capabilities [150].

A subset of these agents is tactically deployed in the cloud, their activities deftly choreographed by orchestration agents. Concurrently, other agents are strategically positioned at the network’s edge, ensuring localized domain functionality in close proximity to the end user. This judicious deployment and orchestration of agents furnishes the requisite flexibility and scalability to assure seamless network management. Moreover, it imparts adaptability to the system, facilitating accommodation of shifts in the network milieu [160].

Furthermore, the architecture pledges the delivery of high-quality service to end-users by providing an adept mechanism to monitor and manage the network in an efficient and effective manner. This innovative architecture fundamentally underpins the reliable, adaptive, and high-performing network management system we propose, ensuring optimized network operations in dynamic conditions [117].

Figure 3.3: MANA-NMS simulation based on the multi-agent system. Adapted from [150].
3.4 Strategizing Agent Deployment: A Crucial Aspect of Network Management

The effective deployment and association of agents within an agent-based network system present a significant challenge, given the intricate nature of the task. A crucial technique employed in this context is agent chaining, an approach that involves executing a series of agents in a carefully curated sequence to ensure seamless service delivery to the end user.

Consider, for instance, the task of predicting incoming traffic. This prediction can facilitate the instantiation of requisite resources or agents. Subsequently, a traffic classifier agent can discern the type of incoming traffic, thus setting the stage for the establishment of an agent chain for service processing aligned with the stipulations of the service level agreement (SLA). This methodology allows the delivery of highly differentiated services.

This approach is elucidated in Figure 3.4, which presents an exemplar of service chaining as a tangible proof of concept. In this depicted scenario, a range of agents, such as traffic classification agents, traffic predictors, QoS agents, database management agents, routing agents, and orchestration agents, are tactically deployed to constitute an agent chain for service processing. This implementation demonstrates the profound potential of agent chaining in creating flexible, adaptive, and highly effective network management systems [150].

![Figure 3.4: Agents Chaining. Adapted from [150].](image-url)
3.5 Addressing Real-world Challenges through Advanced Agent and System Architecture

Faced with the prevalent challenge of an absence of an encompassing framework and an appropriate simulation environment for the real-world assessment of intelligent multi-agent network systems, we have designed an innovative approach using our comprehensive internal agent and system architecture [151].

To accomplish this objective, we have assimilated three distinct scenarios as agents within our system: a network traffic classifier, a predictor, and a Quality of Service (QoS) agent. The integration of these agents has culminated in a dynamic, adaptable network management system capable of effectively responding to a wide range of network conditions and user demands.

Our system’s versatility and dynamism, resulting from this innovative combination, ensure that it can anticipate, identify, and swiftly respond to a broad spectrum of scenarios that could emerge in the complex landscape of modern network management. This adaptability makes the system uniquely equipped to accommodate evolving network conditions and user needs, thereby enhancing its overall utility.

Following meticulous testing and evaluation, we have demonstrated that our novel approach can significantly enhance network performance, improve scalability, and enable effective service delivery. Consequently, our innovative system architecture presents a promising solution to the multifaceted challenges faced in the realm of modern network management.

3.5.1 Optimizing Network Performance with the Intelligent Network Traffic Classifier Agent

Let us denote the raw data set obtained from the Preprocessing Agent as $X$, with $Y$ representing the corresponding labels. The role of the Network Traffic Classifier Agent,
as elaborated in Algorithm 1, can be outlined as a sequence of systematic steps. The Network Traffic Classifier Agent applies various computational strategies to classify network traffic. These strategies include data preprocessing, computation of class weights, splitting the dataset, training the model, and saving and plotting the model. The time complexity of each of these functions significantly influences the performance of the agent. The in-depth time complexity analysis of the Training Agent class is presented in Appendix A. This analysis provides insights into the performance of the agent and can aid in the optimization of future iterations of the system.

The initial task involves dividing $X$ and $Y$ into training and validation subsets, a crucial step to ensure the model’s robustness and capacity for generalization (Step 1) [79]. Following this, the model architecture must be meticulously defined. An Artificial Neural Network (ANN) or Convolutional Neural Network (CNN) may serve as a suitable choice due to their inherent ability to capture underlying data patterns and structures (Step 2) [81].

The subsequent phase involves training the model by employing the divided data, corresponding labels, and class weight. This process aims to optimize the model’s parameters (Steps 3 and 4) [58]. The trained model’s performance is then evaluated using the validation subset to assess its ability to generalize learned patterns to unseen data (Step 5).

Upon completion of model evaluation, the results are compiled and displayed (Step 6). Simultaneously, the trained model is stored in a versatile format (such as h5) to support future implementation [40]. The classifier’s outcome is then encapsulated in a message containing encoded labels, class weight, and length of labels (Step 7). This message is then relayed to the Predictor Agent, enabling it to utilize the trained model for making subsequent predictions (Step 8).

Following this systematic procedure, the Network Traffic Classifier Agent can effectively segregate network traffic data, enabling the Network Traffic Predictor Agent to
make precise predictions. The end result is an enhanced network performance, leading to a significantly improved user experience.

Algorithm 1 Network Traffic Classifier Agent behavior.

Require: $X$ from Preprocessing agent, $Y$ Labels
Ensure: Encoded Labels $E$, Class weight $w$, nb_class $n$, length of Labels $l$

☞ Step 1 Split $X$ and $Y$ for validation (ensure model robustness).
☞ Step 2 Define model architecture (e.g., ANN, CNN).
☞ Step 3 Train model with split data, $Y$, and $w$.
☞ Step 4 Optimize model parameters with training data.
☞ Step 5 Evaluate model with validation data.
☞ Step 6 Print evaluation outcomes, save model (e.g., in h5).
☞ Step 7 Create message $M$ with classifier’s outcome.
☞ Step 8 Send $M$ to Predictor Agent for future predictions.

3.5.2 Intelligent Network Traffic Predictor Agent behavior

The efficacy of intelligent network systems hinges heavily on the accuracy of their predictions, which directly impacts the quality of network management and end-user experience. The Network Traffic Predictor Agent lies at the heart of such systems, leveraging pre-trained models to predict future network behavior, thereby informing and optimizing the decision-making processes.

Let $D_t$ represent the test dataset produced by the Network Traffic Classifier Agent (Algorithm 1), and $M$ symbolize the trained model. The Network Traffic Predictor Agent applies $M$ to $D_t$ to anticipate network traffic behavior (Algorithm 2). This process can be formulated as follows:

Given $D_t$ and $M$, the Predictor Agent executes the following steps to predict network traffic behavior:

- Load the trained model $M$.
- Employ $M$ to formulate predictions on $D_t$. 

44
• Analyze the predicted results using evaluation metrics such as confusion matrix, precision, recall, and F1 score to ascertain the model’s accuracy and reliability [140].

• Print and archive the prediction performance metrics, such as loss and accuracy, to establish a baseline for continuous improvement.

• Compile a message summarizing the Predictor Agent’s outcome, offering key insights on model performance and prediction accuracy.

• Forward the message to the QoS Agent, facilitating informed decision-making for network management.

By accurately predicting traffic behavior, the Predictor Agent enables proactive and informed decision-making, leading to enhanced network performance, more efficient resource allocation, and improved user experience.

\[\text{Algorithm 2 Network Traffic Predictor Agent behavior.}\]

\begin{algorithm}
\textbf{Require:} Test Dataset \(D_t\), Trained model \(M\) (Algo. 1)
\textbf{Ensure:} Predicted results \(P\)
\begin{algorithmic}
\State \(\text{Step 1} \quad P \leftarrow M(D_t):\) Predict network traffic using the trained model \(M\) on test dataset \(D_t\).
\State \(\text{Step 2} \quad \) Analyze \(P\) with evaluation metrics, e.g., confusion matrix \(C\).
\State \(\text{Step 3} \quad \) Format \(C\) for visualization and interpretation.
\State \(\text{Step 4} \quad \) Store and print prediction performance metrics, i.e., loss \(L\) and accuracy \(A\).
\State \(\text{Step 5} \quad \) Compile message \(M_s\) summarizing predictor’s outcome.
\State \(\text{Step 6} \quad \) Forward \(M_s\) to QoS Agent for subsequent decision-making.
\end{algorithmic}
\end{algorithm}

3.5.3 Performance Optimization via Intelligent QoS Monitoring and Provisioning Agent

As an integral component of our agent-based network system, the QoS (Quality of Service) Monitoring and Provisioning Agent’s role is critical. Responsible for preserving the
network’s service quality, this agent capitalizes on the insights gained from the Network Traffic Predictor Agent (Algorithm 2) and actual network throughput measurements. For a comprehensive understanding of the agent’s behavior, refer to Algorithm 3.

The crux of this problem is outlined as follows: given a dataset $D$ representing network traffic, an assortment of possible machine learning models $M$, and sets $T$ and $V$ denoting possible partitions for training and validation data respectively, the aim of the QoS Monitoring and Provisioning Agent is to pinpoint the most suitable machine learning model $m \in M$ in tandem with corresponding partitions $(d_t, d_v) \in T \times V$. This combination should optimize the model’s performance when analyzing network traffic data.

Specifically, the steps undertaken are:

- Divide the dataset $D$ into mutually exclusive training and validation sets, i.e., $D = d_t \cup d_v$ with $d_t \cap d_v = \emptyset$.

- Leverage a Window Generator object for comprehensive data segmentation, aiding efficient analysis [58].

- Instantiate an optimal model $m \in M$, the selection of which is contingent upon the specific data type and task requirements.

- Employ the training set $d_t$ to train the selected model $m$, represented as $m \leftarrow \text{train}(m, d_t)$.

- Evaluate the model’s proficiency using the validation set $d_v$ through performance($m, d_v$).

- Document and present the model’s performance metrics, including accuracy and loss.

- Compile a message that encapsulates the QoS outcome.

- Transmit this message to the Database Agent, aiding in the subsequent ranking of paths for end-to-end service routing between given source-sink pairs.
With these steps, the QoS Monitoring and Provisioning Agent, by astutely selecting the optimal model and data partitions, ascertains an optimized network performance, thereby ensuring an elevated quality of service for end users [42].

**Algorithm 3** QoS Monitoring and Provisioning Agent behavior.

**Require:** Predictor Agent Information (Algo. 2), Network throughput measurements

**Ensure:** Data windowing for performance analysis

☞ **Step 1** Partition the Data Set into training ($D_{\text{train}}$) and validation ($D_{\text{valid}}$) subsets

☞ **Step 2** Configure a Window Generator $WG$ for data segmentation

☞ **Step 3** Instantiate a suitable model $m \in \{\text{Linear, Dense, CNN, LSTM}\}$

☞ **Step 4** Train $m$ on $D_{\text{train}}$

☞ **Step 5** Evaluate $m$ on $D_{\text{valid}}$

☞ **Step 6** Print metrics $\mu(m, D_{\text{valid}})$

☞ **Step 7** Create QoS outcome $Q$

☞ **Step 8** Send $Q$ to Database Agent for path ranking

3.6 Case Studies: Paradigm Shifts in Network Service Management

**Leveraging Intelligent Agents**

The metamorphosis of network architectures in the last decade signifies a noteworthy technological leap, largely due to the integration of intelligent systems and the strategic utilization of agents as fundamental building blocks. Notwithstanding these profound advancements, there persists a lacuna in the provision of adequate simulation environments to thoroughly assess the performance of these multi-agent systems within such network architectures [18], [150], [161].

In light of this, our groundbreaking research endeavors to bridge this gap by formulating and implementing a diverse range of network service agents leveraging the avant-garde Python Agent DEvelopment (PADE) framework [103], [151]. This innovative project is set to redefine how we perceive and evaluate the efficacy of intelligent systems, offering a highly sophisticated experimental environment designed to examine an extensive array
of machine learning and deep learning algorithms.

The open-source nature of our project facilitates broad accessibility, allowing any interested parties to delve into the intriguing realm of intelligent systems within the network architecture. The project’s implementation can be accessed through the following GitHub link: https://github.com/pmushidi2/MasIntNetSys.git. The simulation environment is architectured to enable an array of intelligent algorithm experiments, making it an exemplary platform for researchers and industry mavens aiming to forge novel technologies and systems that could spearhead the future of network architectures.

The provision of this experimental environment is envisaged to catalyze the evolution of more robust, adaptive, and efficient intelligent network architectures capable of tackling the progressively complex demands of modern networks. The potential of this endeavor is profound, and we look forward to the significant opportunities it holds for researchers, scholars, and industry professionals across the globe.

3.6.1 Leveraging PADE’s Potential: Streamlining Multi-Agent System Implementation

The role of PADE (Python Agent DEvelopment) framework in the successful actualization of our proposed multi-agent system is instrumental. Embodying a robust, flexible, and open-source software framework, PADE is explicitly engineered to streamline the development, execution, and management of multi-agent systems in distributed computing ecosystems. What particularly distinguishes PADE is its unmatched accessibility and user-centric design, facilitated by its foundation on the versatile Python programming language [56].

Equipped with a comprehensive array of functionalities such as asynchronous message passing, intuitive agent creation, and convenient remote execution, PADE fosters automation in system operations. These capabilities render PADE an exceptional choice for the design and implementation of sophisticated multi-agent systems, reiterating its
significance in fostering efficient and high-performing multi-agent system implementa-

tion.

In our endeavor to provide a holistic perspective of available multi-agent system (MAS) platforms, we present Table 3.2, a comprehensive delineation of leading MAS platforms in the contemporary market. The table encompasses vital information, such as the primary domain and programming language of each platform, thereby empowering researchers and practitioners with necessary insights for an informed platform selection. By offering this comparative information, we enable users to perform a thorough analysis of the various alternatives, ultimately guiding them to the platform that seamlessly aligns with their project’s unique requirements.

3.6.2 Synergizing PADE with MAS Architecture for Advanced Network Management

Our groundbreaking framework, as visualized in Figure 3.5, strategically coalesces with PADE’s state-of-the-art architecture to fully exploit its comprehensive capabilities. The bedrock of our technology landscape is founded within the hardware layer, encompassing an assortment of devices capable of hosting mainstream operating systems such as Linux, Windows, or MacOSX [75]. To enhance accessibility and user-friendliness, we have incorporated an open-source Python distribution (CPython 3.6 or higher) into these devices. The device range includes a multitude of machines from Raspberry Pis to diverse embedded system development boards that leverage ARM-based processors and operate on Linux [152].

The intermediate layer of our architecture is fortified by PADE modules, which introduce a Python-Twisted support stratum. This layer facilitates the design and operation of multi-agent applications, dependent on their respective libraries and environment, thus forming the backbone of any intricate multi-agent application. The Agent Management System (AMS), an integral component, supervises the overarching operation of the PADE
Table 3.2. Development of Multi-Agent Systems: A Survey of Platforms.

<table>
<thead>
<tr>
<th>Platform</th>
<th>Domain</th>
<th>Language</th>
<th>Open Source</th>
<th>Intelligent Network</th>
<th>Ref.</th>
</tr>
</thead>
<tbody>
<tr>
<td>OpEMCSS</td>
<td>Complex systems</td>
<td>C</td>
<td>No</td>
<td>No</td>
<td>[41]</td>
</tr>
<tr>
<td>MaDKit</td>
<td>Supply chains</td>
<td>Java</td>
<td>Yes</td>
<td>No</td>
<td>[61]</td>
</tr>
<tr>
<td>JADE</td>
<td>Distributed apps</td>
<td>Java</td>
<td>Yes</td>
<td>No</td>
<td>[20]</td>
</tr>
<tr>
<td>NetLogo</td>
<td>Social phenomena</td>
<td>Java</td>
<td>Yes</td>
<td>No</td>
<td>[147]</td>
</tr>
<tr>
<td>JAS</td>
<td>General purpose</td>
<td>Java</td>
<td>Yes</td>
<td>No</td>
<td>[141]</td>
</tr>
<tr>
<td>Swarm</td>
<td>Computer simulation</td>
<td>Obj-C</td>
<td>Yes</td>
<td>No</td>
<td>[107]</td>
</tr>
<tr>
<td>RePast</td>
<td>Social Science</td>
<td>Multiple</td>
<td>Yes</td>
<td>No</td>
<td>[43]</td>
</tr>
<tr>
<td>MASON</td>
<td>Discrete event</td>
<td>Java</td>
<td>Yes</td>
<td>No</td>
<td>[91]</td>
</tr>
<tr>
<td>AnyLogic</td>
<td>Hybrid system</td>
<td>Java</td>
<td>Yes</td>
<td>No</td>
<td>[28]</td>
</tr>
<tr>
<td>OsBrain</td>
<td>General purpose</td>
<td>Python</td>
<td>Yes</td>
<td>No</td>
<td>[76]</td>
</tr>
<tr>
<td>PADE</td>
<td>General purpose</td>
<td>Python</td>
<td>Yes</td>
<td>No</td>
<td>[103]</td>
</tr>
<tr>
<td><strong>Our Framework</strong></td>
<td><strong>Complex Network</strong></td>
<td>Python</td>
<td>Yes</td>
<td>Yes</td>
<td>[150]</td>
</tr>
</tbody>
</table>

platform, orchestrating agent creation, deletion, and migration [21].

At the apex of our architecture, agents such as classifiers, predictors, databases, and Quality of Service (QoS) confluence to fulfill the designated tasks. By harnessing the potential of PADE architecture, our framework is poised to deliver high-performance, fault-resilient, and distributed multi-agent systems capable of seamlessly processing extensive data volumes. Our innovative construct presents a formidable platform for the development of intelligent networks, enabling real-time, data-driven decision-making processes.
The seamless integration with PADE’s avant-garde architecture unlocks the multi-agent systems’ full potential, empowering them to navigate complex challenges across diverse domains. Consequently, our technology extends a new paradigm for multi-agent systems that can catalyze researchers and developers to engineer sophisticated and innovative applications.

Figure 3.5: MAS-Architecture framework integrated with PADE: The bottom layer consists of the hardware on which our framework runs. The middle layer is supported by PADE modules, providing a Python-Twisted support layer for multi-agent applications. The top layer is the application layer, where agents such as classifiers, predictors, databases, and QoS collaborate to accomplish tasks.

3.6.3 Towards Optimal Multi-Agent Systems: Comprehensive Evaluation

Metrics for Performance and Efficiency

Multi-agent systems, a central element in distributed artificial intelligence, are best evaluated through the lens of the Quality of Service (QoS) they deliver. QoS, as the keystone metric, provides a holistic overview of system efficiency by assessing several key aspects including latency, accuracy, decision error rates, power consumption, and computational complexity. Ascertaining these metrics furnishes invaluable insights into the operational efficacy of the system, its capacity for providing timely and accurate results,
cost-effectiveness, and crucially, its potential for sustainability in an increasingly energy-conscious world.

Power consumption, a metric of burgeoning importance in multi-agent systems, merits particular attention. The cumulative energy efficiency of each agent invariably impacts the system’s overall power demand. Given the escalating focus on the global energy footprint, diligently monitoring and optimizing power consumption not only minimizes operational costs but also enhances the system’s sustainability profile, ensuring its continued viability in a resource-constrained future.

In tandem, computational complexity presents another crucial metric when assessing multi-agent systems. Highly intricate systems demand significant resources for operation and maintenance, thus necessitating their optimization. Striking a balance between system complexity and cost-effectiveness is a delicate yet essential task that ensures the longevity and economic viability of the system.

Communication latency, too, is an influential factor in the performance of a multi-agent system. The time it takes for agents to exchange information can directly impact system output, and reducing this latency can significantly enhance overall system performance. Rapid and efficient information exchange is crucial to creating a cohesive, high-functioning multi-agent system that maximizes output and minimizes overhead.

A comprehensive evaluation of multi-agent systems encompasses a wide array of metrics, with QoS, power consumption, computational complexity, and communication latency leading the charge. Meticulous tracking and optimization of these metrics pave the path towards improved system performance and efficiency. This ensures that multi-agent systems continue to deliver unparalleled service, thereby catering to their users in the most effective manner possible.
3.6.4 Delving Deeper: A Comprehensive Examination of Case Study Results

In the realm of this research, we endeavored to emulate a real-world network scenario to thoroughly appraise the efficacy of our novel Multi-Agent System (MAS)-based intelligent network platform. The outcomes drawn from the meticulous analysis of latency, accuracy, and loss statistics have lent a compelling testament to the exemplary performance of our system. A granular examination of our results unveiled the remarkable performance of three key actors within our MAS - the preprocessing agent, the classifier agent, and the predictor agent (refer to Table 3.3).

The preprocessing agent (NTPrA) emerged as a crucial player, making significant strides in data transformation and readiness, as corroborated by our case study outcomes. Through its function of cleaning, transforming, and reducing high-dimensionality data, it facilitated improved decision-making in subsequent agent processes.

Likewise, the classifier agent (NTCA) displayed an exceptional ability to categorize new, unseen instances into predefined classes. Through the use of state-of-the-art machine learning algorithms, the classifier agent maintained high accuracy levels, showcasing a robust model with superior generalization capabilities.

The predictor agent’s performance (NTPA) was notable. Leveraging advanced forecasting models, the predictor agent accurately anticipated future events, thereby enabling preemptive actions and enhancing the overall responsiveness of our system.

Table 3.3. Effect of Data Set Size on Performance of MAS Using ANN Architecture

<table>
<thead>
<tr>
<th>T</th>
<th>Size (MB)</th>
<th>NTPrA</th>
<th>NTCA</th>
<th>NTPA</th>
<th>Accuracy</th>
<th>Loss</th>
</tr>
</thead>
<tbody>
<tr>
<td>30 m.</td>
<td>1</td>
<td>1.64secs</td>
<td>791.45secs</td>
<td>2.79secs</td>
<td>88.63%</td>
<td>0.3621</td>
</tr>
<tr>
<td>1H</td>
<td>5.9</td>
<td>1.76secs</td>
<td>9705.16secs</td>
<td>6.41secs</td>
<td>89.3%</td>
<td>0.3334</td>
</tr>
<tr>
<td>1H30</td>
<td>6.6</td>
<td>1.74secs</td>
<td>1792.58secs</td>
<td>9.67secs</td>
<td>96%</td>
<td>0.1491</td>
</tr>
<tr>
<td>2H</td>
<td>12.8</td>
<td>1.81secs</td>
<td>12266.58secs</td>
<td>15.5 secs</td>
<td>91.89%</td>
<td>0.2601</td>
</tr>
</tbody>
</table>
Our system’s performance was bolstered significantly by the inclusion of the Prolog Agent Communication Language (PADE) ACL, a ground-breaking technology that fosters real-time network monitoring. The PADE ACL serves as the backbone of our system, enabling swift identification and resolution of potential issues, marking an evolution in proactive network management. Moreover, the facility to gauge variable intra-agent communication delays provides insights that could be leveraged for optimizing system performance and resource management.

Our study highlighted an intriguing trend - as the duration of data collection increased, so did the communication time between agents. This underscores the delicate equilibrium between the duration of data collection and the agents’ real-time processing capabilities, suggesting the need for an adaptive mechanism to maintain optimal efficiency. Persistent data collection can inadvertently burden the agents, leading to increased communication delays and thereby diminishing the system’s overall performance (Table 3.4). The implications of these findings shed light on the importance of intelligent data collection strategies, reinforcing the need to strike a balance between data richness and system efficiency.

Table 3.4. Agent Communication Latency in Seconds.

<table>
<thead>
<tr>
<th>Duration</th>
<th>Hours</th>
<th>NTPrA-NTCA</th>
<th>NTCA-NTPA</th>
<th>NTPA-NTQoSA</th>
</tr>
</thead>
<tbody>
<tr>
<td>After 30 mins</td>
<td>0.5</td>
<td>2.21</td>
<td>9.53</td>
<td>8.08</td>
</tr>
<tr>
<td>After 1 hour</td>
<td>1</td>
<td>7.25</td>
<td>9.77</td>
<td>15.76</td>
</tr>
<tr>
<td>After 1.5 hours</td>
<td>1.5</td>
<td>15.51</td>
<td>10.58</td>
<td>28.57</td>
</tr>
<tr>
<td>After 2 hours</td>
<td>2</td>
<td>19.06</td>
<td>11.86</td>
<td>34.53</td>
</tr>
</tbody>
</table>

The integration of a state-of-the-art Quality of Service (QoS) agent (NTQoSA) armed with cutting-edge neural network algorithms has led to significant enhancements in our ability to predict and mitigate network congestion on a user’s network path. This potent combination aids in maintaining optimal network performance levels by providing actionable insights ahead of potential disruptions, marking a significant leap forward in
preemptive network management strategies.

The efficacy of this strategy is further confirmed by our comparative analysis of various throughput prediction models. The results underscore the superior capabilities of our multi-agent system (MAS) based intelligent network approach in enhancing network performance. This innovative MAS-based framework distinguishes itself from traditional models by demonstrating a superior ability to handle network complexity, making it an ideal solution for the evolving demands of modern networks.

![Figure 3.6: Mean Absolute Error of different models on Validation and Test data sets.](image)

Table 3.5 illustrates the extraordinary predictive prowess of the Long Short-Term Memory (LSTM) model, which was employed to predict the aggregate link throughput. The LSTM model consistently achieved impressive accuracy in forecasting within random one-hour intervals. In this context, links denote a variety of network associations between two hosts within the network, identified based on previously collected network data. Such associations can include a wide spectrum of connections such as router-router, router-switch, and switch-computer connections, among others, underscoring the model’s versatility in handling diverse network configurations.
The performance analysis, conducted on identical datasets, reveals that each successive model architecture within the Multi-Agent System (MAS)-based intelligent network approach significantly supersedes its predecessor in performance. This highlights the efficacy of our MAS-based intelligent network approach in amplifying network performance. This innovative approach delivers a resilient and adaptive solution for network optimization, facilitating proactive identification and swift resolution of network issues to ensure optimal performance. The adaptive nature of the MAS-based approach allows it to handle the evolving complexities of modern network systems, making it a robust solution for next-generation networks.

Table 3.5. Network Throughput Prediction: Accurate Results at 1-Hour Intervals Using LSTM.

<table>
<thead>
<tr>
<th>1 h Pred</th>
<th>link-1</th>
<th>link-2</th>
<th>link-3</th>
<th>link-4</th>
<th>link-5</th>
<th>link-6</th>
<th>link-7</th>
<th>link-8</th>
<th>link-9</th>
<th>link-10</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>17.3</td>
<td>26.1</td>
<td>5.3</td>
<td>34.2</td>
<td>42.7</td>
<td>33.7</td>
<td>66.8</td>
<td>29.0</td>
<td>2.9</td>
<td>6.8</td>
</tr>
<tr>
<td>2</td>
<td>17.8</td>
<td>29.3</td>
<td>15.6</td>
<td>23.0</td>
<td>36.5</td>
<td>29.2</td>
<td>82.2</td>
<td>33.5</td>
<td>3.3</td>
<td>5.8</td>
</tr>
<tr>
<td>3</td>
<td>21.3</td>
<td>14.6</td>
<td>7.0</td>
<td>26.8</td>
<td>39.2</td>
<td>15.4</td>
<td>84.9</td>
<td>34.0</td>
<td>3.0</td>
<td>5.2</td>
</tr>
<tr>
<td>4</td>
<td>20.4</td>
<td>35.7</td>
<td>4.7</td>
<td>31.0</td>
<td>36.3</td>
<td>33.6</td>
<td>127.8</td>
<td>35.4</td>
<td>2.4</td>
<td>5.3</td>
</tr>
<tr>
<td>5</td>
<td>21.3</td>
<td>32.1</td>
<td>4.9</td>
<td>39.1</td>
<td>39.3</td>
<td>31.1</td>
<td>127.2</td>
<td>32.7</td>
<td>2.6</td>
<td>5.6</td>
</tr>
<tr>
<td>6</td>
<td>19.5</td>
<td>39.0</td>
<td>5.6</td>
<td>20.9</td>
<td>257.6</td>
<td>26.9</td>
<td>80.2</td>
<td>34.0</td>
<td>2.3</td>
<td>5.5</td>
</tr>
</tbody>
</table>

The amalgamation of advanced neural network algorithms, notably the LSTM model, into the Multi-Agent Systems (MAS)-based intelligent network paradigm imparts profound insights into network operations. These insights, more than just understanding, provide an actionable diagnosis of network ailments, enabling prompt and accurate remedial actions for seamless and efficient network operation. This inventive approach thus offers a vanguard solution to the rapidly escalating intricacies of network optimization, carving a well-defined trajectory for future research and development in this compelling domain.

This investigation underscores the potency of MAS-based intelligent network platforms in delivering seamless and reliable user experiences. It establishes that the MAS
approach is not merely a theoretical concept but a practical tool capable of transforming network performance. The study also unveils an expansive opportunity landscape for further optimization and enhancement of network performance, potentially triggering a new wave of advancements in the field of intelligent network systems.

3.7 Chapter Summary

This chapter delivers an in-depth analysis of the revolutionary design and optimization of Softwarized Intelligent Network Architecture, powered by multi-agent systems (MAS), presenting a groundbreaking approach to efficient, reliable, and adaptive network systems. Commencing with a comparative exploration of traditional monolithic and modern microservice architectures, it elucidates the superiority of the latter in terms of modularity, adaptability, and maintenance ease.

However, the primary focus is on the pivotal shift to agent-based systems, highlighting their advanced dynamism, adaptability, and decentralized control. Despite acknowledging the increased design complexity and security considerations, the chapter underscores agent-based systems’ potential in creating sophisticated, intelligent networks.

A core discussion involves the application of MAS in designing next-generation networks, exhibiting how agent intelligence can enable proactive response, resilience, and self-healing mechanisms. In this vein, it introduces an innovative methodology for intelligent network architecture development, leveraging software-based agents as a transformative strategy in network research.

The chapter further dissects the advanced internal architecture of agents and a sophisticated system architecture featuring a combination of cloud and edge-deployed agents, promoting flexibility and scalability. It also introduces agent chaining as an efficient solution for network management and demonstrates a novel framework to address the lack of a comprehensive simulation environment for multi-agent network systems.

Detailing the function of key agents such as the Network Traffic Classifier, Predictor,
and the QoS Monitoring and Provisioning Agent, the chapter corroborates their efficiency through testing and evaluation, displaying a significant improvement in network performance and service delivery.

Another major contribution is the illustration of a groundbreaking project that harnesses the Python Agent Development (PADE) framework to redefine the application of intelligent agents in network architectures. Emphasizing the necessity of key metrics tracking, the chapter presents an examination of case study results that showcase the performance of various agents within the MAS-based intelligent network paradigm.

The chapter concludes by underscoring the superiority of the LSTM model in predictive capabilities, validating the MAS-based approach as a practical tool for network performance transformation. The resilience and adaptability of the MAS-based approach, with each model architecture surpassing its predecessor, signify a wealth of opportunities for future network optimization and enhancements.
Chapter 4

SE-DO Framework: Boosting Intelligent Agent Efficiency in Resource-Limited IoT Networks

The burgeoning prevalence of interconnected IoT devices brings with it a host of complex management challenges. These challenges are amplified by the need to create intelligent and autonomous networks that can engage in self-configuration, self-healing, and self-management. While the adoption of intelligent agents presents a promising pathway towards this end, their dynamic management demands considerable data processing for training network function agents. This poses significant strains on resource-limited environments, such as IoT devices, thereby calling for innovative solutions.

The core of this chapter revolves around a groundbreaking approach known as Scalable and Efficient DevOps (SE-DO) [19], designed to optimize the performance of intelligent agents in these resource-constrained environments. This SE-DO framework is founded on a robust multi-agent system architecture, which promotes both reactive responses and proactive anticipation in reconfiguring network systems to meet dynamic requirements.

Given the forthcoming advances in next-generation networks like 6G, that will necessitate hyper-efficient, reliable solutions, the SE-DO approach is of particular significance. In these networks, a myriad of new services and applications will be enabled, thereby amplifying the need for efficient resource management.

In this chapter, we delve into the successful implementation of SE-DO in a multi-agent system, exploring the roles and impacts of various machine learning models, including ANN, CNN, and RNN, on agent performance within resource-constrained contexts. We draw from experiments with real-world data to highlight the high accuracy and efficiency delivered by our proposed architecture, even within the limitations of resource-constrained environments.
Through this exploration, we aim to demonstrate the real-world applicability and scalability of the SE-DO approach, underscoring its potential as a game-changing solution for the optimization of intelligent agents in future network systems.

4.1 Transforming Telecommunication Landscape: Harnessing Multi-Agent Strategies for 6G Networks and Beyond

The continuous evolution of the telecommunications industry has been significantly influenced by the progressive introduction of 5G networks. Currently, the relentless call for all-encompassing connectivity and advanced applications is forging the path toward the arrival of 6G networks. These forthcoming networks hold the promise of fundamentally altering our interface with technology. Serving as a precursor to unparalleled innovations, 6G networks promise to surpass their 5G predecessors in terms of phenomenal speeds, reduced latency, and markedly expanded functionalities. They are poised to provide reinforced security protocols, unmatched reliability, and compatibility with emergent technologies like immersive media and the Internet of Things (IoT) [38], [83], [169].

As the domain of interconnected devices continues to widen and the intricacies of communication patterns escalate, the conventional microservice-based design is grappling with its own set of limitations. The lack of necessary flexibility to accommodate the dynamic needs of next-generation networks gives rise to various obstacles [148]. This intrinsic rigidity can lead to inefficient resource utilization, scalability concerns, and an overall decline in network performance. Consequently, it’s imperative to advocate for innovative design strategies capable of overcoming these challenges and unlocking the full potential of 6G networks.

The solution to these predicaments necessitates a more adaptable and dynamic strategy. A multi-agent-supported modular design strategy emerges as a promising solution, offering numerous benefits such as enhanced flexibility, adaptability, and fault tolerance,
which culminates in an efficient and scalable network system.

Moreover, this agent-oriented approach is fundamental to intelligent network design. Considering the swift rise in traffic behavior and diverse application demands in next-gen networks, a network design that is resilient and adaptable is crucial.

In-network intelligence, grounded on machine and deep learning algorithms, serves as the cornerstone of multi-agent-based network management systems in future networks like 6G. However, deploying intelligent agents in resource-limited environments poses a formidable challenge [15]. In response to this, we suggest the Scalable and Efficient Development Operations (SE-DO), a holistic approach aimed at optimizing intelligent agent performance in resource-restricted environments [63].

Although prior studies have proposed network architectures for next-gen networks [64], [120], a significant discrepancy exists between theoretical constructs and practical applications. Our work endeavors to bridge this gap by introducing the SE-DO methodology for implementing a multi-agent intelligent network system in IoT-constrained environments. This groundbreaking approach offers considerable contributions to the field of networking, facilitating the establishment of robust networks capable of accommodating the growing demands of emerging services and applications.

Our proposed method is well-positioned to reconcile the theory-practice disparity and provides a prospective solution for developing intelligent agent network systems in IoT-constrained environments. This method bears significant potential, paving the way for future research and advancements in the field of networking.

4.2 Intelligent Agents in Resource-Constrained Environments: Challenges and Innovative Solutions

The integration of intelligent agents in resource-constrained environments, such as Internet of Things (IoT) networks, poses formidable challenges arising from limited CPU, memory, and storage capacities, as well as restrictions in data collection, storage, and
communication latency. Pioneering research has dedicated efforts to overcome these obstacles, with a primary focus on optimizing computational, storage, and communication resources while maintaining high agent performance.

A pivotal challenge is the scarcity of computational resources available to these devices. To address this, Kang et al. [77] devised groundbreaking deep compression techniques for neural networks, achieving a delicate balance between resource consumption and agent performance. Complementary to this, He et al. [65] proposed an efficient channel pruning method to eliminate redundant channels in convolutional neural networks, effectively conserving computational resources.

Storage capacity constraints represent another critical impediment, but innovative solutions have emerged. Han et al. [64] introduced a deep compression method, while Wen et al. [159] proposed weight quantization techniques, both successfully reducing storage requirements without compromising agent performance.

Overcoming limitations in communication bandwidth has been achieved through pioneering techniques, exemplified by Lee et al.’s [82] compression method and Lin et al.’s [88] weight quantization approach, which effectively mitigate bandwidth demands.

Hurdles concerning data collection, storage, and latency have been tackled by Mao et al. [100] with a data compression method and Wang et al. [156] with a distributed training approach, both contributing to reduced storage demands and training latency.

Integration of multi-agent systems and machine learning techniques holds the promise of enhancing network performance [151] [13], but introduces inherent challenges, including increased complexity and potential underperformance. To address these concerns, streamlining cognitive components, enhancing computational efficiency, and employing adaptable decision-making frameworks emerge as compelling strategies. Machine learning offers enhanced decision-making capabilities, while integration of fault-tolerant features and advanced data analytics techniques elevates accuracy and decision-making precision.

Deploying intelligent agents within resource-constrained environments necessitates
innovative solutions. Promising strategies found in existing literature can significantly boost network performance. In this work, we present "SE-DO," an innovative and comprehensive approach that addresses the entire ecosystem, striking a harmonious equilibrium between the reliability of data collection and analytic agents, the accuracy of training and prediction agents, and the constraints of the environment. SE-DO offers practical and scalable solutions, paving the way for substantial performance enhancements across diverse applications.

4.3 SE-DO: Scalable & Adaptive Agent Deployment in Resource-Constrained Environments

SE-DO, an innovative and highly scalable method is designed to seamlessly integrate additional agents as per the evolving objectives of the system. Our architecture features data collection agents that securely compile data, while preprocessing agents refine it for subsequent analysis or training stages. Training agents utilize this preprocessed data to optimize machine learning models, and predictor agents generate real-time predictions. This dynamic and efficient architecture ensures a high level of accuracy, efficiency, and security within defined resource constraints, making SE-DO a promising candidate for next-generation network environments.

Our groundbreaking approach adeptly addresses resource constraints, consistently achieving superior levels of accuracy, efficiency, and security. SE-DO remains perpetually adaptive to fluctuating network conditions, ensuring continuous updates and peak performance of the intelligent network system. As illustrated in Figure 4.1 (SE-DO Workflow), our methodology employs continuous data collection, analysis, training, and prediction to maintain an advanced system. Emphasizing a seamless process for developing and deploying intelligent agents within resource-constrained environments, SE-DO’s key components enable ceaseless development, deployment, and monitoring of intelligent agents. By strategically positioning agent types throughout the SE-DO lifecycle, we en-
sure effective operations within the constraints of resource-limited environments.

4.3.1 Dynamic Agent Lifecycle Orchestration for Resource-Constrained Environments

In this novel approach for orchestrating the lifecycle of intelligent agents in resource-constrained environments, our methodology comprises four pivotal agent types, each strategically positioned throughout the lifecycle to optimize development, deployment, and monitoring processes.

The Data Collector Agent operates within the Continuous Integration (CI) phase, ensuring seamless integration and testing of code changes. Swift identification and rectification of potential issues streamline the development process. In the Continuous Delivery (CD) phase, the Data Analytics or Preprocessing Agent orchestrates an automated deployment process tailored for resource-constrained environments, guaranteeing reliability, efficiency, and scalability amidst the ever-evolving digital landscape. Positioned in the Infrastructure as Code (IaC) phase, the Data Training Agent utilizes microservices to oversee agent deployment, leveraging the advantages of treating infrastructure as code.
for simplified management and scalability. Lastly, the Data Predictor Agent, situated in the Monitoring and Logging phase, enables real-time performance tracking of individual agents and the overall system. Leveraging logs and metrics, this agent promptly identifies and resolves operational issues, enhancing system efficiency and effectiveness.

Our dynamic agent lifecycle orchestration presents a pioneering solution for navigating resource limitations while optimizing the performance and adaptability of intelligent agents in diverse applications.

The integration of an array of specialized agents, inclusive of Quality of Service (QoS) agents, is facilitated through the SE-DO model, making it an all-encompassing strategy for network administration in up-and-coming networks such as 6G. Central to the tenets of the SE-DO framework is the insistence on regular amalgamation and examination of code modifications. This anticipatory measure provides a platform for the timely recognition and resolution of potential hurdles during the developmental phase, guaranteeing a robust and dependable network milieu.

Further, the deployment of intelligent agents, specifically within environments with limited resources, is mechanized by the SE-DO model. Such mechanization lessens manual input, thereby mitigating the likelihood of inaccuracies and leading to a swifter and more effective institution of network management solutions.

Integral to the SE-DO paradigm is its emphasis on the collaboration and communication amongst team members. The achievement of a successful multi-agent system is contingent on effective teamwork. By endorsing the exchange of knowledge, coordination, and synergy, the SE-DO framework cultivates a cooperative workspace that augments the efficacy of network management initiatives.

The SE-DO model is founded on real-time performance tracking and logging. These methods ensure that operational issues are swiftly identified and rectified, thereby guaranteeing ongoing network performance optimization. The mechanization of testing, deployment, and monitoring processes by the model not only minimizes errors but also
streamlines operations and improves overall system efficacy. Such automation allows the system to dynamically scale and adjust to the increasing demands of emerging services and applications.

Through the adoption of the SE-DO model, the development and deployment of intelligent agents in resource-limited settings become a cyclic and iterative process. Such repetition assures scalability, reliability, and efficiency, meeting the stringent requisites of forthcoming networks. Network administrators are provided with an exhaustive framework by the SE-DO model to overcome resource limitations and manage complex network environments effectively. It guarantees the delivery of high-performance network services, satisfying the evolving needs of emerging applications and services.

4.4 A New Hierarchical Architecture for Smart Network Agent Automation

Figure 4.2: Hierarchical Architecture for Smart Network Agent Automation.

Figure 4.2 showcases a cutting-edge hierarchical architecture specifically tailored for network automation agents operating in resource-limited environments. This innovative
design empowers efficient and effective management of complex network intricacies, addressing the unique challenges encountered in such settings.

At the foundation of the architecture lies the Input and Analysis layer, serving as the fundamental pillar for the agent’s operations. Continuously monitoring and collecting real-time network data, this layer provides a comprehensive view of the network environment, granting valuable insights into its status and performance. This data is then seamlessly passed on to the Cognitive Components layer, acting as the intellectual core of the agent. Leveraging sophisticated algorithms and reasoning engines, this layer performs intricate data analysis, extracting valuable insights that facilitate informed decision-making processes.

Moving up the hierarchical structure, the Planning layer exhibits remarkable adaptability and flexibility. Drawing from the insights provided by the Cognitive Components layer, it formulates implementable strategies to optimize network performance. These strategies are intelligently designed to align with the dynamic network conditions, enhancing the agent’s overall efficiency.

The subsequent Configuration layer plays a vital role in generating the necessary configuration files required for network devices to effectively implement the planned strategies. Ensuring the correctness and feasibility of the agent’s decisions before implementation, the Validation layer assumes a crucial position in the architecture. It meticulously verifies the generated configurations, validating their compatibility with the network environment, thereby minimizing the risk of disruptive changes and ensuring precision in the agent’s actions.

As the final layer, Output serves as the indispensable feedback mechanism for the agent’s operations. It establishes seamless communication with external systems, providing crucial information and timely updates on the network’s status and performance. This feedback mechanism allows for smart and effective network management, ensuring prompt adjustments when necessary.
To achieve enhanced precision and efficiency in resource-restricted settings, the architecture incorporates state-of-the-art machine learning techniques throughout its structure. These techniques refine the decision-making and planning processes, enabling the agent to dynamically adapt and respond to evolving network conditions. Additionally, advanced neural network models are seamlessly integrated, augmenting the agent’s performance and further enhancing precision in its operations.

Notably, this comprehensive hierarchical architecture overcomes the limitations of previous multi-agent approaches, as discussed in [151]. It firmly emphasizes the precision and reliability of network automation agents, effectively addressing potential drawbacks encountered in prior methodologies.

Furthermore, the architecture prides itself on ensuring seamless integration and interoperability with other layers and systems. Leveraging interfaces with APIs, messaging protocols, and standard network communication protocols, the agent effortlessly integrates with existing network infrastructure, ensuring smooth communication and unhindered collaboration.

By embracing this advanced hierarchical architecture, network operators can optimize network performance, effectively reduce operational expenses, and bolster security and reliability. Its innovative design and integration of advanced technologies have a profound impact on the field of networking, fostering efficient and intelligent network automation.

4.5 Efficient Strategies for Next-Generation Network Optimization

4.5.1 Data Collection Agent

In the dynamic landscape of next-generation networks, such as 6G, the development of efficient strategies for network management is paramount due to limited resources and ever-changing network states. Key to this endeavor is the data collector agent, a pivotal player responsible for gathering and processing data to enhance intelligent agent performance in these environments.
To address the optimization challenge, several parameters are defined. The dataset, denoted by $X$, must be of sufficient size ($N$) to enable effective analysis. Concurrently, the constraints of storage ($C$), energy ($E$), and computation ($P$) need careful consideration. Additionally, memory limitations ($M$) and the essential information threshold ($I$) play significant roles. The data collection strategy ($S$) adopted must also be well-suited to the network environment.

The overarching objective is to optimize these variables while adhering to the given constraints. By doing so, we can achieve remarkable enhancements in the performance of intelligent agents within the resource-constrained 6G networks. The optimization endeavors encompass the following objectives:

- Maximizing resource utilization to harness available capacities effectively.
- Minimizing energy consumption during data collection and processing to conserve power.
- Ensuring efficient storage management to accommodate the collected data seamlessly.
- Optimizing computation allocation to meet the processing demands efficiently.
- Adhering to memory limitations while preserving essential information.
- Designing effective data collection strategies tailored to the unique network environment.

Algorithm 4 outlines our approach to address these optimization goals. By focusing on these objectives, we can profoundly enhance overall network efficiency and empower intelligent agents to operate optimally within the resource-constrained 6G network landscape. The detailed time complexity analysis of the Efficient Multi-Objective Optimization Algorithm for Network Data Collection is presented in Appendix A. This comprehensive
Algorithm 4 Efficient Multi-Objective Optimization Algorithm for Network Data Collection

Require: Data collector agent $D$
Ensure: Optimized dataset $X_{opt}$
1: Initialize dataset $X$; Define $N, C, E, P, M, I, S$
2: Define metrics $m = \{\text{storage, energy, computation, memory}\}$
   // Process raw data with $D$ to get $X_{raw}$
3: Assign weights $w_{m_i}$ to each $m_i$ in $m$
   // Define multi-objective function
4: $f(X) = \Sigma w_{m_i} * m_i(X)$
   // Define constraints for optimization
5: $C \equiv \{|X| \geq N, \text{storage}(X) \leq C, \text{energy}(X) \leq E, \text{computation}(X) \leq P, \text{memory}(X) \leq M, \text{information}(X) \geq I, \text{strategy}(X) = S\}$
   // Select suitable optimization algorithm
6: $O \leftarrow \text{SelectOptimizationAlgorithm}(f, C)$
   // Minimize objective function with constraints
7: $X_{opt} \leftarrow O.minimize(f(X), C)$
   // Evaluate performance of $X_{opt}$
8: Perf $\leftarrow \text{EvaluatePerformance}(X_{opt}, f)$
   // Visualize results
9: Vis $\leftarrow \text{Visualize}(X_{opt}, \text{Perf})$
10: return $X_{opt}$, Perf, Vis

Analysis furnishes insights into the performance of the system, thereby offering invaluable guidance for optimizing subsequent iterations of the system.

The discussed procedure unveils a tailored optimization methodology conceived explicitly for data gathering agents, aiming at the augmentation of network efficiency. This technique initiates by formulating an uninitialized data structure, denoted as $X$, while simultaneously setting the parameters for optimization. These parameters include $N$, signifying the necessity for data points, $C$ indicating the limitations on data storage, $E$ outlining the boundary for energy consumption, $P$ illustrating the capacity for computation, $M$ showing the restrictions on memory, $I$ denoting the baseline requirement for information, and $S$, signifying the planned approach for data collection [47].

The function $f(X)$ is established as the objective function and symbolizes the aggregate consequences of storage, energy, computational capabilities, and memory constraints on the optimization dilemma. Subsequently, the algorithm lays down the constraints encompassing the obligatory quantity of data points ($N$), the upper limit on storage capacity ($C$), the maximum limit for energy utilization ($E$), the computational capability
ceiling \( P \), the boundary for memory utilization \( M \), the baseline for the information requirement \( I \), and the compulsory conformity to the pre-established data gathering strategy \( S \).

To secure the minimization of the objective function \( f(X) \), while concurrently abiding by the aforementioned constraints, the algorithm implements optimization techniques. Such strategies could include evolutionary algorithms or mathematical programming, offering a broad range of methods to navigate complex solution spaces [90]. The application of an evolutionary algorithm is eloquently illustrated in the context of multi-objective optimization for network data collection, as detailed in Algorithm 5. Upon the completion of these steps, the algorithm procures the optimized dataset \( X \) and yields this as the resulting output [105].

Linear programming, a popular technique in mathematical programming, can be particularly useful in solving complex optimization problems. A linear programming problem can be mathematically formulated as follows:

\[
\text{Minimize: } c^T x, \text{ subject to: } Ax \leq b, x \geq 0
\]

Here, \( x \) is a vector of decision variables, \( c \) and \( b \) are vectors of coefficients, and \( A \) is a matrix of coefficients for the constraints.

Integer programming is a variant of linear programming where all or some of the decision variables must be integers. It’s an appropriate tool when dealing with discrete decision variables. A typical integer programming problem is as follows:

\[
\text{Minimize: } c^T x, \text{ subject to: } Ax \leq b, x \geq 0, x \in \mathbb{Z}^n
\]

Convex optimization is a subset of mathematical programming used when the objective function and the feasible region are both convex. Convex optimization problems can be written as: \textbf{Minimize: } \( f(x) \), \textbf{subject to: } \( g_i(x) \leq 0, i = 1, \ldots, m \), and \( h_i(x) = 0, i = 1, \ldots, p \)

Here, \( f(x) \) is a convex function to be minimized, \( g_i(x) \) are convex inequality constraints, and \( h_i(x) \) are affine equality constraints.
These mathematical programming techniques provide robust methods for tackling optimization problems in 6G networks. By modeling the constraints and objectives mathematically, they allow for precise and efficient optimization.

Evolutionary algorithms exhibit a strong capability for global search, a crucial characteristic in the multifaceted landscape of 6G networks. Their mathematical properties can be analyzed using the Markov Chain model, where the state transition probabilities form a matrix $P = [p_{ij}]$ with $p_{ij}$ representing the probability of transitioning from state $i$ to state $j$. The iterative nature of the algorithm can be captured by the equation $X_{t+1} = PX_t$, where $X_t$ is a vector representing the state of the algorithm at time $t$ [51]. The concept of fitness can be related to the Fundamental Theorem of Natural Selection, expressed as $\frac{dw}{dt} = G(t)$, where $\frac{dw}{dt}$ represents the rate of increase in mean fitness, and $G(t)$ is the genetic variance in fitness at that time [122].

The performance of the evolutionary algorithm can be quantitatively analyzed through time and space complexity, essential metrics for determining the algorithm’s scalability and efficiency. Suppose $T(n)$ represents the time complexity of an algorithm with respect to the input size $n$. The Master Theorem provides a method for determining a "big O" solution to the recurrence relation $T(n) = aT\left(\frac{n}{b}\right) + f(n)$ [44].

Stability of the solution is another fundamental property that should be examined. Let’s denote the state of the system at time $t$ by $X_t$, and $X$ represents a stable state. If we find a Lyapunov function $V(X)$ such that $\Delta V = V(X_{t+1}) - V(X_t) \leq 0$ for $X \neq X$ and $\Delta V = 0$ for $X = X$, we can conclude that $X$ is a stable state under the Lyapunov Stability theory [95].

Altogether, these mathematical formulations highlight the intricate properties of evolutionary algorithms used for optimization in 6G networks. Understanding these mathematical underpinnings is vital for developing efficient and stable optimization strategies.

In the modified version of the algorithm (refer to Algorithm 5), we are initializing a population of potential solutions (Step 9) and then continuously evolving that population.
**Algorithm 5** Evolutionary Multi-Objective Optimization Algorithm for Network Data Collection

**Require:** Data collector agent $D$

**Ensure:** Optimized dataset $X_{opt}$

1: Initialize dataset $X$; Define $N, C, E, P, M, I, S$
2: Define metrics $m = \{\text{storage, energy, computation, memory}\}$
   // Define raw data with $D$ to get $X_{raw}$
3: Assign weights $w_m$, to each $m_i$ in $m$
   // Define multi-objective function
4: $f(X) = \Sigma w_m \times m_i(X)$
   // Define constraints for optimization
5: $C \equiv \{|X| \geq N, \text{storage}(X) \leq C, \text{energy}(X) \leq E, \text{computation}(X) \leq P, \text{memory}(X) \leq M, \text{information}(X) \geq I, \text{strategy}(X) = S\}$
   // Initialize population for EA
6: $P \leftarrow \text{InitializePopulation}(f, C)$
   // Define selection, crossover and mutation operators
7: $S, C, M \leftarrow \text{DefineOperators}()$
   // Run evolutionary optimization algorithm
8: **while** termination criteria not met **do**
9: \[ P_{selected} \leftarrow S.select(P) \]
10: \[ P_{crossed} \leftarrow C.cross(P_{selected}) \]
11: \[ P_{mutated} \leftarrow M.mutate(P_{crossed}) \]
12: \[ P_{new} \leftarrow \text{CreateNewPopulation}(P, P_{mutated}) \]
13: \[ P \leftarrow P_{new} \]
14: **end while**
15: $X_{opt} \leftarrow \text{GetBestSolution}(P)$
   // Evaluate performance of $X_{opt}$
16: $\text{Perf} \leftarrow \text{EvaluatePerformance}(X_{opt}, f)$
   // Visualize results
17: $\text{Vis} \leftarrow \text{Visualize}(X_{opt}, \text{Perf})$
18: **return** $X_{opt}, \text{Perf}, \text{Vis}$

The modified algorithm makes use of evolutionary principles to navigate the complex solution space effectively, maintaining a balance between exploring new areas of the…
space (diversity) and exploiting already discovered promising areas (convergence), which is crucial in multi-objective optimization problems.

4.5.2 The Role of the Data Analytic Agent

The intricacies associated with the task confronting a data analytics or preprocessing entity can be encapsulated as an optimization challenge, defined by a distinct objective function and a set of constraints. To shed light on this perspective, consider the following function and limitations:

\[
\begin{align*}
\text{Minimize} & \quad f(Z) = \sum_{\text{constraint} \in \{Q,F,R,N\}} \text{constraint}(Z) \\
\text{Subject to} & \quad Z = g(X), \quad \text{constraint}(Z) \in \mathcal{K}
\end{align*}
\]

where

\[
\mathcal{K} = \begin{cases} 
Q(Z) \leq \text{storage limit}, \\
F(Z) \leq \text{energy limit}, \\
R(Z) \leq \text{computation limit}, \\
N(Z) \leq \text{memory limit}, \\
\text{knowledge}(Z) \geq K \end{cases}
\]

In this context, \( Z \) symbolizes the collection of preprocessed data, while \( X \) stands for the entire set of data collated by the data collector entity. The various constraints - \( Q, F, R, \) and \( N \) - embody restrictions on storage, energy, computation, and memory respectively. \( K \) signifies the minimum acceptable knowledge content. The function \( g \) performs critical tasks on the collected data, including feature extraction, normalization, sampling, and one-hot encoding.

Optimizing this objective function necessitates the deployment of diverse techniques, such as gradient descent, which can be selected according to data characteristics such as size, complexity, and accuracy, along with available computational resources. The
optimal choice should strike a balance between precision and efficiency while maximizing performance and minimizing associated costs.

The Continuous Analysis and Preprocessing (CAP) principle is profoundly applicable in this scenario. It highlights the necessity of continually analyzing and preprocessing data in real-time, ensuring its relevancy and contemporaneity. By incessantly fine-tuning the data collection and preprocessing mechanism, the system can yield more accurate and efficient datasets, which can be used to educate deep learning models, such as Convolutional Neural Networks (CNNs), Artificial Neural Networks (ANNs), and Recurrent Neural Networks (RNNs) [98], [123].

The suggested strategy, which amalgamates feature extraction, normalization, sampling, and one-hot encoding, can significantly enhance the quality of the dataset for training deep learning models. Consequently, intelligent networks are empowered to make more precise and insightful decisions.

4.5.3 The Role of the Data Training Agent

The focus of our work, as detailed in Algorithm 6, involves an innovative strategy for training a designated model employing preprocessed data, whilst adhering to an array of optimization constraints. Our methodology is deeply rooted in the SE-DO framework, paving the path for continuous education and adaptability of intelligent entities within emerging network generations like 6G. With SE-DO as our guiding light, the algorithm we propose strives to curtail the objective function, taking into consideration the limitations imposed by storage, energy utilization, computation, and memory. The ultimate ambition is to train the chosen model to attain a high degree of accuracy, while optimizing resource utilization.

Let’s initiate by letting $X$ represent the parameters of the model, $y$ signify the output variable, and $D$ embody the preprocessed dataset. The objective function that requires minimization is designed as such:
Minimize $\psi \ f(\psi)$

Subject to $Y = f(X), \ \text{constraint}(\psi) \in \mathcal{C}$

where $\mathcal{C} = \left\{ \begin{align*}
\text{storage}(\psi) & \leq C_1, \\
\text{energy}(\psi) & \leq C_2, \\
\text{computation}(\psi) & \leq C_3, \\
\text{memory}(\psi) & \leq C_4
\end{align*} \right\}$

In this equation, $\psi$ denotes the parameters of the chosen model, and $g(\psi)$ is the objective function that requires minimization. The boundaries include the maximum permissible values for storage, energy consumption, computational requirements, and memory utilization.

To tackle this optimization problem, we exploit cutting-edge techniques and the \texttt{prob.solve()} methodology. The variables in the optimization problem are refreshed with the calculated values and we sketch the total cost relative to the information content. By perpetually educating the model and evaluating its performance, the agent’s adaptability to ever-changing network conditions is ensured, thereby maximizing its utility over time.

\textbf{Algorithm 6} Training Agent Behavior

\begin{algorithmic}
\Require Training Data $\mathcal{D}$
\Ensure Trained Model $\mathcal{M}$, Optimization Results $\mathcal{R}$
\begin{enumerate}
\item Define \texttt{train\_model} function $f_m$
\item Partition $\mathcal{D}$ into training and testing sets $(\mathcal{D}_{\text{train}}, \mathcal{D}_{\text{test}})$
\item Train Sequential model $\mathcal{M} = f_m(\mathcal{D}_{\text{train}})$
\item Log training $(\ell_{\text{train}})$ and validation metrics $(\ell_{\text{val}})$ at final epoch
\item Compute info content $(I)$, energy $(E)$, computation $(C)$, and memory $(M)$ values
\item Solve optimization problem $\mathcal{R} = \text{prob.solve}(I, E, C, M)$
\item Update optimization variables with $\mathcal{R}$
\item Plot total cost $(TC)$ against information content $I$
\end{enumerate}
\end{algorithmic}

Algorithm 6 offers an overview of the Data Training Agent’s behaviour within the SE-DO framework. Through Algorithm 6 and the SE-DO methodology, we validate
the potency of continuous training and adaptation of the chosen model to attain high performance while adhering to various constraints. By upholding the SE-DO principle, our methodology guarantees that the Data Training Agent operates as an astute and nimble entity in the dynamic 6G network environment. Applying the same logic as mentioned above, we can formulate efficient designs for other agents within the SE-DO framework, such as the Data Predictor Agent and Quality of Service Agent, as elaborated in [150].

4.5.4 The Role of the Data Predictor Agent

The predictive paradigm articulated in Algorithm 7 deploys an optimization scaffold, aspiring to hone the predictive precision of a pre-trained machine learning archetype, concurrently factoring in several restrictions such as energy expenditure, computational capability, and memory utilization. This predictive construct leverages an agent-oriented optimization methodology that encompasses the resolution of an optimization challenge aimed at attenuating the expenditure whilst augmenting the content of information.

This modus operandi demands as inputs a trained machine learning prototype denoted as $M$, and both the inputs $X_{test}$ and the outputs $y_{test}$ of the testing data, and consequently resolves the ensuing optimization quandary:

Minimize $\theta$

Subject to

\[ Y = f(X), \quad \text{constraint} (\psi) \in \mathcal{C} \]

where

\[ \mathcal{C} = \begin{cases} 
\text{storage}(\theta) \leq C_1, \\
\text{energy}(\theta) \leq C_2, \\
\text{computation}(\psi) \leq C_3, \\
\text{memory}(\theta) \leq C_4 
\end{cases} \]
The optimization formulation delineated above portrays a methodology to judiciously select $\theta$, a parameter that is intrinsic to the framework we are examining. This process aims to minimize the holistic function signified by $\text{cost}(\theta)$. This objective function embodies the integral costs affiliated with the operational performance of a predictive model, inclusive of computational consumption, energy expenditure, memory requirements, and the valuable information content the model offers.

Evidently, this optimization scenario is subject to constraints, represented within the set $C$. These boundaries comprise upper limits on storage (limited to $C_1$), energy utilization (bounded by $C_2$), computational load (not exceeding $C_3$), and memory engagement (capped at $C_4$).

This optimized approach ensures the judicious use of resources, making it a prudent and practical choice for computational models. A major goal of this methodology is to strike a balance between the overall performance of the predictive model and the resources consumed, thus presenting an optimal scenario that ensures both operational efficiency and resource prudence.

**Algorithm 7** Data Predictor Agent Behavior

<table>
<thead>
<tr>
<th>Require: Trained model $M$, Test data inputs $X_{test}$, outputs $y_{test}$</th>
</tr>
</thead>
</table>
| 1: procedure EVALMODEL  
Evaluate $M(X_{test}, y_{test})$  
Display MAE, accuracy |
| 2: end procedure |
| 3: procedure PREDICT($X_{test}, y_{test}$)  
$\hat{y}_{test} \leftarrow M(X_{test})$  
Record prediction time $t$  
Minimize: $E(M), P(M), M(M)$ under constraints  
Update variables of the optimization problem  
Plot results  
Output: $\hat{y}_{test}, E(M), P(M), M(M)$, total cost, $t$  
Plot $\hat{y}_{test}$ vs $y_{test}$ |
| 4: end procedure |

The outlined algorithm commences its operation by gauging the performance of the pre-trained model, denoted by $M$, on the test data variables $X_{test}$ and their corresponding outputs $y_{test}$. This is accomplished by computing key performance metrics including
The core of the algorithm is the optimization problem it tackles to minimize resource usage while ensuring the constraints pertaining to energy, computational capabilities, and memory allocation are all adhered to. This careful balance ensures the most efficient use of resources while maintaining model performance.

The solution of this optimization problem provides the optimal values for energy usage, computational power, and memory allocation. The algorithm then applies these values to update the variables of the optimization problem, further refining the model’s efficiency and performance.

These operations culminate in a set of visualizations that effectively communicate the results, including a graph plotting the trade-off between resource usage and model performance. In addition to the graphical representation, the algorithm provides a detailed output of key metrics, such as the predictor values, energy consumption, computational power requirements, memory usage, overall cost, and prediction time, offering a comprehensive overview of the model’s performance and resource efficiency. A final plot presents a side-by-side comparison of the predicted and actual values of $y_{test}$, illustrating the model’s predictive capability.

The core strength of the algorithm lies in its ability to maintain high prediction accuracy while optimizing resource usage, making it a valuable tool in resource-constrained environments, such as edge computing or mobile devices.

4.6 Enhancing AI Performance in Next-Generation Networks: An Exploration of the SE-DO Framework

In the subsequent part of this discourse, we highlight the efficacy of our innovative approach, the System-Enhanced Data-Optimized (SE-DO) methodology, in amplifying the
performance of AI-powered entities, especially those functioning within the constraints of future networking paradigms. This schema propounds the uninterrupted and distributed execution of an exhaustive array of data-centric operations ranging from gathering and examination to model training and prediction, thereby promoting efficient data handling.

Recognizing the collaborative spirit of scientific inquiry and the growing trend of open-source practices in AI research, we have made our implementation code publicly available. To this end, we invite interested scholars and industry professionals to delve into our source code, available in our GitHub repository. We believe that such unrestricted access to our work will stimulate exploration and inspire further enhancement of our methodology.

The SE-DO methodology, we believe, embodies the ideal combination of efficiency, adaptability, and transparency, marking a significant step forward in the realm of AI-optimized networks. The aim is not only to foster a better understanding and appreciation of our work but also to encourage its application in diverse and potentially unforeseen contexts.

4.6.1 Deciphering the Fabric of Network Topology

The experimental framework we’ve developed, utilizing the capabilities of NetSim version 13.3, is adept at capturing the intricate nature of 6G network configurations as presented in Figure 6.4. A series of Raspberry Pi units [102], assigned the role of data-gathering entities, constantly amass real-time information, thereby allowing for network flexibility. An exhaustive tabulation of the key features of Raspberry Pi 4 Model B is compiled in Appendix B. This tabulation provides comprehensive insight into the system’s capabilities and can serve as a vital reference point for utilizing and enhancing future iterations of this computational platform. In a virtual environment facilitated by NetSim, this data undergoes a transformation to generate a credible 6G network representation. A series of analytical tools scrutinize this virtual model to derive measures of network performance,
while an in-built learning component progressively enhances the predictive accuracy of the system. The insights generated by the predictive agent serve as invaluable metrics on network response to varied traffic conditions.

Our intricate system finds its foundation in the robust OsBrain framework [151], a platform known for its effectiveness in fostering distributed computing setups through fluid agent interactions via message exchanges. Our work sheds light on the significant benefits of utilizing multi-agent structures, which are realized via Docker images as an alternative to conventional micro-service strategies.

Our innovative network system, implemented within a Docker environment [46] and harnessing a multitude of smart agents, utilizes the RESTful API for seamless inter-agent communication, offering several benefits over traditional centralized message-passing frameworks such as OsBrain. The RESTful API, a respected standard in the field of component communication, simplifies the integration of a variety of systems and intelligent agents. Its stateless protocol feature further permits system expansion horizontally, free from the dependency on a centralized broker that characterizes message-passing architectures. Additionally, the application of conventional security protocols like SSL/TLS guarantees data protection and consistency in RESTful API. Its allowance for loose component coupling further eases development and maintenance processes, a sharp contrast to the message-passing systems’ requirement for agent interdependence.

Even in the absence of a formal worldwide 6G network specification as of September 2023, the beneficial attributes of our Docker-enabled multi-agent intelligent network model can be anticipated to constitute potential aspects of future 6G networks, which are still in their developmental stages. The Postman application oversees the management and testing of API calls between agents and performance monitoring, while Docker’s containerized structure permits agent operation across varied machines or clusters, promoting effective resource usage and fault tolerance. Consequently, the Docker-based simulation environment emerges as a scalable and efficacious solution for developing sim-
ulation systems with extensive real-world applicability.

Our method’s innate scalability, resilience, and emphasis on data security, achieved through integrating a micro-service approach via OsBrain, underscores the inherent advantages and efficacy of our model. This enables an in-depth understanding and subsequent optimization of 6G network performance. Consequently, our novel network design can adapt and evolve in sync with real-world network conditions, offering a highly precise, comprehensive, and adaptable medium for exploring and boosting 6G network operations.

Figure 4.3: An illustrative depiction of the 6G network experimental setup integrating multiple Raspberry Pi devices, NetSim simulation, and OsBrain multi-agent configurations.

4.6.2 Dataset

In the development of our experimental blueprint, we leveraged a data repository emanating from a dynamic, self-adjusting network configuration, as illustrated in Figure 6.4, that mirrors the complexity inherent in 6G networks. The stream of information, incessantly flowing from a consortium of Raspberry Pi gadgets functioning as data ac-
quisition entities, represents real-time network conditions with an unprecedented level of fidelity. These components meticulously document the oscillations within the network, thereby becoming indispensable elements in the fabrication of an authentic 6G network representation within the NetSim virtual platform.

An assortment of network parameters is assembled from the harvested information, encapsulating rudimentary data such as data packet dimensions, in addition to integral metrics including training and validation deficits, precision in training and validation, the mean deviation in prognoses, and total prognostic precision. The comprehensiveness of this data repository is vital for enhancing the precision and reliability of the subsequent analysis and optimization stages, thereby contributing significantly to the overall efficacy of our proposed model.

4.6.3 Scalable, Resilient, and Efficient: Unveiling the Performance of Multi-Agent Systems in 6G-IoT Networks through the SE-DO Framework

Leveraging the robustness of the SE-DO architecture, our study introduces a smart, multi-agent system, painstakingly designed to oversee critical operations: data gathering, manipulation, model training, and inference. This systematic allocation of specialized roles is executed through the launch of four expert agents, each empowered by state-of-the-art machine learning methodologies and tailored for optimal efficiency and performance.

Firstly, the Data Collector Agent, enhanced by the capabilities of the Scapy library [92], works assiduously to pull out key attributes from network packets. Paired with the Pulp library, it is set to tackle complex optimization tasks, unwaveringly targeting a reduction in data acquisition costs while maintaining completeness. The Data Analytic Agent assumes the significant duty of improving the quality of the harvested data. It conscientiously expunges missing entries, converts non-numeric fields into digestible for-
mats, and standardizes features using Z-score normalization. Furthermore, it employs one-hot encoding to handle categorical elements, thus ensuring data homogeneity.

Next, the Model Training Agent harnesses the robust methodologies of Artificial Neural Networks (ANNs), Convolutional Neural Networks (CNNs), and Recurrent Neural Networks (RNNs). Through the strategic employment of dropout regularization, it balances dataset dimensions and classifications, fostering an environment conducive to machine learning. Lastly, our Model Evaluation Agent functions as a rigorous appraiser of our trained machine-learning models. It is entrusted with anticipating future data trends while assiduously working towards the minimization of energy, computational resources, and memory utilization. It strictly adheres to the restrictions related to data gathering and information content, reflecting its effectiveness.

Collectively, these agents yield a spectrum of optimization outcomes and performance indicators, markedly enhancing data governance efficiency and providing valuable insights into the effectiveness of our machine-learning models. The strength of this multi-agent system lies in its robustness, scalability, and operational efficiency. It demonstrates remarkable performance under the rigorous conditions presented by a 6G network.

The Data Collector Agent remains ever-alert, continually adapting its operations to maintain an optimal balance between data acquisition and resource usage, thus ensuring timely updates. The Model Training Agent displays adaptability to data variations, continually enhancing model precision and evolutionary potential. The Model Evaluation/Predictor Agent provides real-time model performance analysis, reacting promptly to any changes.

Blessed with intrinsic flexibility and adaptability, our multi-agent system exhibits resilience to the rising demands of future network generations. It sets a new standard in intelligent networking, thus positioning itself as a compelling benchmark in the forthcoming 6G era.

Our investigation targets the issues related to the monumental data influx triggered by
interconnected Internet of Things (IoT) devices and the intricate nature of forthcoming network generations, such as 6G. Proposing SE-DO (Scalable and Efficient DevOps), a proficient and expandable solution, this study aims to enhance the efficacy of smart agents operating within IoT realms that are innately resource-limited. We scrutinized the effectiveness of three divergent model architectures (Convolutional Neural Networks, Artificial Neural Networks, and Recurrent Neural Networks) across a spectrum of packet sizes within two distinct deployment tactics: service-micro based and agent-centered.

### 4.6.4 Scalability Analysis

Scalability in a system refers to its inherent capability to uphold or even elevate performance when supplemented with additional resources such as data or memory. Pertaining to our Multi-Agent System (MAS), the models exhibited profound scalability traits. As data packet size expanded, the performance of models surged, as observed in both service-micro and agent-centric approaches, as illustrated in Tables 4.1 and 4.2. This performance enhancement was discernible in several metrics, including a downturn in training and validation losses, and a climb in accuracies.

Specifically, the agent-oriented approach surpassed the micro-service approach consistently, primarily when data packet sizes swelled. This implies the superior scalability inherent in the agent-based method, promising potential advantages for MAS deployments that need to manage more substantial data loads. Similarly, prediction precision and mean absolute error (MAE) exhibited advancement as packet sizes rose, further substantiating the scalability of the proposed approaches (as shown in Figure 4.4).
### Table 4.1. Scalability and Resilience Analysis (Micro-services-based Strategy)

<table>
<thead>
<tr>
<th>Packet Count</th>
<th>Training</th>
<th>Validation</th>
<th>Prediction</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Loss</td>
<td>Accuracy</td>
<td>Loss</td>
</tr>
<tr>
<td>Convolutional Neural Network (CNN)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>95000</td>
<td>0.05274</td>
<td>94.84%</td>
<td>0.21355</td>
</tr>
<tr>
<td>97000</td>
<td>0.04561</td>
<td>95.52%</td>
<td>0.09330</td>
</tr>
<tr>
<td>99000</td>
<td>0.06112</td>
<td>94.00%</td>
<td>0.12308</td>
</tr>
<tr>
<td>120000</td>
<td>0.07109</td>
<td>92.95%</td>
<td>0.08470</td>
</tr>
<tr>
<td>180000</td>
<td>0.04139</td>
<td>95.93%</td>
<td>0.08191</td>
</tr>
<tr>
<td>Artificial Neural Network (ANN)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>95000</td>
<td>0.00967</td>
<td>96.94%</td>
<td>0.00199</td>
</tr>
<tr>
<td>97000</td>
<td>0.00361</td>
<td>97.74%</td>
<td>0.00083</td>
</tr>
<tr>
<td>99000</td>
<td>0.00834</td>
<td>97.72%</td>
<td>0.00195</td>
</tr>
<tr>
<td>120000</td>
<td>0.00340</td>
<td>97.71%</td>
<td>0.00090</td>
</tr>
<tr>
<td>180000</td>
<td>0.00478</td>
<td>98.31%</td>
<td>0.00079</td>
</tr>
<tr>
<td>Recurrent Neural Network (RNN)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>95000</td>
<td>0.01311</td>
<td>97.38%</td>
<td>0.01392</td>
</tr>
<tr>
<td>97000</td>
<td>0.00166</td>
<td>97.82%</td>
<td>0.00007</td>
</tr>
<tr>
<td>99000</td>
<td>0.00558</td>
<td>97.77%</td>
<td>0.00768</td>
</tr>
<tr>
<td>120000</td>
<td>0.00190</td>
<td>97.71%</td>
<td>0.00296</td>
</tr>
<tr>
<td>180000</td>
<td>0.01582</td>
<td>98.35%</td>
<td>0.01857</td>
</tr>
</tbody>
</table>

4.6.5 In-depth Analysis of Data Collection Agents: Comparative Assessments on Resource Allocation, Energy Utilization, and Cost Dynamics

Drawing upon the data gathered and presented in Tables 4.4 and 4.3, and exemplified beautifully in Figure 5.4, several pertinent observations can be deduced, thereby steering the decision-making process related to the allocation of resources, enhancement of energy efficiency, optimization of memory requirements, application of processing power, time management, and cost considerations tied to data gathering. The methodology built on
Table 4.2. Scalability and Resilience Analysis (Agent-based Strategy)

<table>
<thead>
<tr>
<th># Packets</th>
<th>Training</th>
<th>Validation</th>
<th>Prediction</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Loss</td>
<td>Acc.</td>
<td>Loss</td>
</tr>
<tr>
<td>CNN</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>95,000</td>
<td>0.00121</td>
<td>99.87%</td>
<td>0.06397</td>
</tr>
<tr>
<td>97,000</td>
<td>0.03561</td>
<td>94.52%</td>
<td>0.19330</td>
</tr>
<tr>
<td>99,000</td>
<td>0.00038</td>
<td>99.96%</td>
<td>0.00194</td>
</tr>
<tr>
<td>120,000</td>
<td>0.00063</td>
<td>99.93%</td>
<td>0.05037</td>
</tr>
<tr>
<td>180,000</td>
<td>0.00019</td>
<td>99.98%</td>
<td>0.07667</td>
</tr>
<tr>
<td>ANN</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>95,000</td>
<td>0.02536</td>
<td>97.46%</td>
<td>0.03088</td>
</tr>
<tr>
<td>97,000</td>
<td>0.01720</td>
<td>98.27%</td>
<td>0.02947</td>
</tr>
<tr>
<td>99,000</td>
<td>0.01048</td>
<td>98.95%</td>
<td>0.00699</td>
</tr>
<tr>
<td>120,000</td>
<td>0.04550</td>
<td>95.44%</td>
<td>0.06378</td>
</tr>
<tr>
<td>180,000</td>
<td>0.06261</td>
<td>93.73%</td>
<td>0.04846</td>
</tr>
<tr>
<td>RNN</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>95,000</td>
<td>0.17721</td>
<td>82.27%</td>
<td>0.00601</td>
</tr>
<tr>
<td>97,000</td>
<td>0.12990</td>
<td>87.82%</td>
<td>0.00112</td>
</tr>
<tr>
<td>99,000</td>
<td>0.19988</td>
<td>87.77%</td>
<td>0.00259</td>
</tr>
<tr>
<td>120,000</td>
<td>0.17397</td>
<td>90.71%</td>
<td>0.00020</td>
</tr>
<tr>
<td>180,000</td>
<td>0.19612</td>
<td>96.25%</td>
<td>0.00152</td>
</tr>
</tbody>
</table>

Micro-services manifests a more pronounced resource allocation, as exemplified by the greater quantities of accumulated data, energy usage, memory deployment, and processing capabilities.

In contrast, the Agent-based strategy showcases superior prowess in optimizing energy use, thereby curtailing the overall consumption of energy. Additionally, the Agent-based methodology demonstrates significant benefits in limiting memory necessities, leading to more efficient memory management.

While the data-gathering duration showcases negligible variation between the two techniques, it is important to highlight that the Micro-services-centric method stipulates greater processing capabilities. Most notably, the monetary implications associated with
Figure 4.4: Accuracy comparison between micro-services-based and agent-based strategies using different networks: CNN, ANN, and RNN.

data gathering are substantially escalated when the Micro-services-oriented technique is employed, leading to increased expenditure.

Such insights pave the way for well-rounded decision-making, granting stakeholders the discretion to either underscore resource allocation and versatility or to hone in on the twin facets of energy efficiency and cost containment, subject to their individual targets and operational limitations.

Table 4.3. Performance of Data collector agent $f(x)$ for different data set sizes (Agent-based).

<table>
<thead>
<tr>
<th># Packets Stored (MB)</th>
<th>Energy (J)</th>
<th>Memory (MB)</th>
<th>Proc. Power (FLOPS)</th>
<th>Time (min)</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>95,000</td>
<td>6.43</td>
<td>$1.00 \times 10^{-6}$</td>
<td>5.36</td>
<td>$1.00 \times 10^{-6}$</td>
<td>10.01</td>
</tr>
<tr>
<td>97,000</td>
<td>4.64</td>
<td>$1.99 \times 10^{-6}$</td>
<td>3.87</td>
<td>$1.99 \times 10^{-6}$</td>
<td>19.97</td>
</tr>
<tr>
<td>99,000</td>
<td>6.08</td>
<td>$9.97 \times 10^{-7}$</td>
<td>5.07</td>
<td>$9.97 \times 10^{-7}$</td>
<td>9.97</td>
</tr>
<tr>
<td>120,000</td>
<td>9.13</td>
<td>$9.99 \times 10^{-7}$</td>
<td>7.61</td>
<td>$9.99 \times 10^{-7}$</td>
<td>9.99</td>
</tr>
<tr>
<td>180,000</td>
<td>24.34</td>
<td>$9.98 \times 10^{-7}$</td>
<td>20.28</td>
<td>$9.98 \times 10^{-7}$</td>
<td>9.98</td>
</tr>
</tbody>
</table>

4.6.6 Communication Delays Analysis between Agents

An in-depth examination of inter-agent communication delays, as denoted in microseconds ($\mu$s), within a RESTful API framework, is rendered in Table 4.5 and Figure 4.6.
Table 4.4. Performance of Data collector agent $f(x)$ for different data set sizes (Microservices-based).

<table>
<thead>
<tr>
<th># Packets Stored (MB)</th>
<th>Energy (J)</th>
<th>Memory (MB)</th>
<th>Proc. Power (FLOPS)</th>
<th>Time (min)</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>95,000</td>
<td>3.42</td>
<td>$7.99942 \times 10^{-6}$</td>
<td>2.85</td>
<td>$7.99942 \times 10^{-6}$</td>
<td>7.99</td>
</tr>
<tr>
<td>97,000</td>
<td>8.65</td>
<td>$1.39997 \times 10^{-5}$</td>
<td>7.21</td>
<td>$1.39997 \times 10^{-5}$</td>
<td>13.99</td>
</tr>
<tr>
<td>99,000</td>
<td>7.98</td>
<td>$1.30002 \times 10^{-5}$</td>
<td>5.99</td>
<td>$1.30002 \times 10^{-5}$</td>
<td>13.00</td>
</tr>
<tr>
<td>120,000</td>
<td>12.10</td>
<td>$1.50005 \times 10^{-5}$</td>
<td>10.08</td>
<td>$1.50005 \times 10^{-5}$</td>
<td>15.00</td>
</tr>
<tr>
<td>180,000</td>
<td>46.57</td>
<td>$2.20019 \times 10^{-5}$</td>
<td>38.81</td>
<td>$2.20019 \times 10^{-5}$</td>
<td>22.00</td>
</tr>
</tbody>
</table>

Figure 4.5: Performance of Data Collector Agents $f(x)$ for different data set sizes. The left graph shows the performance of micro-services-based, while the right graph shows the performance of agent-based.

Such microsecond-level metrics form the bedrock of our understanding towards optimizing the efficiency of message exchanges to curb the latency in agent-centric architectures. A nuanced observation of the tabulated results uncovers an array of delay durations dispersed across diverse agent pairs and varying packet magnitudes.

The outcome of this rigorous investigation carries a wealth of implications for both academia and industry. They can harness these crucial insights to devise potent methodologies aimed at amplifying the performance of communication processes, curtailing latencies, and thereby enhancing the robustness and overall productivity of agent-oriented systems.
By capitalizing on this knowledge, we can set the stage for future innovations that will accelerate the evolution of smart systems. This will also smoothen the path for impeccable communication amongst agents functioning across a broad spectrum of fields.

Table 4.5. Communication delay via RESTful API (in µs) between different agents.

<table>
<thead>
<tr>
<th># Packets</th>
<th>NTCA-NTPA</th>
<th>NTPA-NTTA</th>
<th>NTTA-NTPrA</th>
</tr>
</thead>
<tbody>
<tr>
<td>95,000</td>
<td>151.35301</td>
<td>12.66590</td>
<td>160.91880</td>
</tr>
<tr>
<td>97,000</td>
<td>247.63489</td>
<td>13.17512</td>
<td>308.56169</td>
</tr>
<tr>
<td>99,000</td>
<td>60.78569</td>
<td>70.35865</td>
<td>103.19917</td>
</tr>
<tr>
<td>120,000</td>
<td>156.12565</td>
<td>170.17319</td>
<td>195.26143</td>
</tr>
<tr>
<td>180,000</td>
<td>63.92751</td>
<td>72.51221</td>
<td>105.33057</td>
</tr>
</tbody>
</table>

Figure 4.6: Communication delay via RESTful API (in µs) between different agents.

Here, NTCA stands for Network Traffic Classifier Agent, NTPA for Network Traffic Preprocessing Agent, NTTA for Network Traffic Training Agent, and NTPrA for Network Traffic Predictor Agent.

4.7 Chapter Summary

In this comprehensive chapter, we introduce the Scalable and Efficient DevOps (SE-DO), an avant-garde approach envisioned to fortify the performance of intelligent agents in IoT networks with resource constraints. Amid the escalating management complexities arising due to an exponential increase in IoT devices, the chapter emphasizes the
pressing need for autonomous, intelligent networks that can self-manage, self-heal, and self-configure.

The groundbreaking SE-DO approach, underpinned by a resilient multi-agent system architecture, is posited as a transformative solution offering reactive responses and proactive anticipation. The chapter strongly advocates its adoption in the face of imminent advancements such as 6G that demand hyper-efficient, reliable solutions. The chapter provides an extensive analysis of various machine learning models, such as ANN, CNN, and RNN, employed within the multi-agent system and assesses their impacts on the performance of intelligent agents, backed by real-world data experiments.

The complex telecommunication landscape, especially the transition from 5G to 6G networks, is examined, pointing out the limitations of traditional microservice-based designs and suggesting a dynamic, multi-agent-supported modular design strategy. The integration of machine and deep learning algorithms for future networks is elucidated, followed by a discussion on the challenges and solutions related to the integration of intelligent agents in resource-limited environments.

Subsequently, the SE-DO framework is portrayed as a practical, all-encompassing strategy for optimizing network management in emerging systems such as 6G. The chapter further details SE-DO’s unique architecture, its adaptivity to fluctuating network conditions, and its emphasis on fostering a collaborative environment and swift resolution of operational issues.

The chapter unfolds an innovative optimization strategy tailored for data collection agents in next-generation networks, encapsulating strategic objectives such as maximizing resource utilization and minimizing energy consumption within an objective function, \( f(X) \). A variety of optimization techniques, including evolutionary algorithms and mathematical programming, are explored in depth.

The role of data analytic agents and the potential of viewing their tasks as optimization challenges are highlighted. The strategy emphasizes the Continuous Analysis and
Preprocessing (CAP) principle as key to maintaining data relevancy and accuracy.

The development of a Data Training Agent and a Data Predictor Agent, both integral to the proposed system, is detailed. These agents, influenced by the SE-DO framework, ensure continuous learning, adaptability, and efficient network management in evolving network generations like 6G while considering resource limitations.

Finally, the chapter presents the SE-DO framework and its implementation, a publicly accessible strategy to enhance AI performance within emerging network architectures. We describe our experimental setup and methodology for data acquisition, the establishment of an efficient multi-agent system, and the evaluation of the SE-DO framework across diverse model architectures. We analyze resource allocation, energy utilization, memory requirements, processing power, time management, and cost dynamics associated with data gathering, ultimately establishing the foundation for future innovations in AI-enhanced networks within the 6G network framework.
Chapter 5

Quality of Service (QoS) and Radio Resource Management (RRM)

The proliferation of Internet of Things (IoT) devices and increasingly data-intensive applications heralds an unprecedented era in communication networks. As we transition from 5G to Beyond 5G (B5G) and 6G networks, the need to manage the escalating volume of traffic and ensure an optimal user experience is paramount. In this context, the themes of Quality of Service (QoS) and Radio Resource Management (RRM) hold the potential to shape the future of network intelligence.

This chapter delves into the profound role of QoS and RRM in the evolving landscape of communication networks, more specifically within the ambit of B5G and 6G technologies. It critically examines the integration of Artificial Intelligence (AI) and Machine Learning (ML) techniques in developing automated, self-aware networks capable of effectively managing resources, mitigating network congestion, and enhancing network security.

We will further delve into the pivotal role of QoS in ensuring seamless user experiences, with an emphasis on traffic-aware predictive responses to maintain a high quality of service, even amid unpredictable traffic demands. The necessity for intelligent network design strategies, employing microservices or multi-agent-based architectures, will also be discussed.

We will introduce a novel proposal for an intelligent QoS agent. By utilizing historical data on user network usage, this agent aims to preemptively identify potential network congestion, recalibrate network paths, and optimize QoS for specific services.

This exploration aims to augment our understanding of current and emerging technologies, and their potential in creating a truly automated, user-centric, and efficient future communication network. Through this study, we hope to contribute to the ongo-
ing discussions and advancements in network intelligence and automation, thereby paving the way for the next-generation B5G and 6G networks.

This cutting-edge research and the development of the proposed QoS and RMM agent could not have been achieved without the invaluable collaboration between our team, the University of New Mexico’s IoT Lab, and the esteemed researchers from the Technical University of Dresden, Germany.

5.1 Advancing Towards Intelligent Networking: A Proposal for a Proactive QoS Agent in Beyond 5G and 6G Networks

As we venture into the era of next-generation communication infrastructures, surpassing the capabilities of 5G and venturing into Beyond 5G (B5G) and 6G, the critical challenge lies in catering to the exponential surge in IoT devices and applications, each with its rigorous requirements [14], [78]. With multiple standardization bodies taking an active role in charting the future of these networks, a defining characteristic of 6G emerges the pervasive integration of Artificial Intelligence (AI) and Machine Learning (ML) techniques for intelligent networking.

AI and ML are not merely ancillary tools but integral elements shaping the fabric of these networks, empowering them to be self-regulating and minimizing the need for human intervention. ML, in particular, is gaining ground in numerous networking tasks, ranging from traffic prediction and resource management to network congestion mitigation and enhancing Quality of Service (QoS) and Quality of Experience (QoE). Moreover, enhancing network security is another domain where ML is proving instrumental.

As the vision of designing adaptive networks with self-regulating, self-healing, self-optimizing, and self-securing capabilities intensifies, it is envisaged that full-scale network automation will be a defining feature of 6G technology [14]. Nonetheless, the quest for achieving comprehensive automation demands constant progress in numerous networking disciplines.
In this evolving scenario, managing QoS and QoE emerges as a critical frontier needing extensive research. An automated system, capable of adapting to user needs, in sync with the often erratic traffic demand, is imperative [145]. A traffic-aware, predictive response system becomes a necessity for fulfilling user QoS/QoE requirements in forthcoming network infrastructures like B5G and 6G. Hence, it is vital for future network designs and technologies to address these challenges head-on, ensuring optimal, automated traffic control, and anticipating future traffic patterns informed by historical data.

Current network design strategies are gravitating towards employing decoupled microservices or multi-agent-based service-oriented architectures [110], [150]. These microservice or agent-based functions can be contained and deployed across a distributed edge or cloud environments. Despite notable progress in traffic prediction and traffic classification agent design, the field of intelligent QoS agent development, especially for autonomous provisioning and control of QoS for specific services, remains largely unexplored.

We put forth the concept of a QoS agent, capitalizing on historical user network usage data within a campus network to predict potential network congestion. The proposed agent, by evaluating the predicted congestion, has the ability to determine alternative paths and reconfigure the optimal path using available path assessments. In essence, we strive to devise a QoS agent capable of assessing available paths and assigning alternative routes to users in alignment with their unique needs [11].

5.2 AI in Networking: Neural Networks for Traffic Forecasting

Established forecasting methods for time series data, such as autoregressive techniques (e.g., AR, ARIMA) along with other conventional methodologies, are abundant. However, they often falter when predicting time, as the requisite information might not be readily available during the execution of the task or might be procured belatedly. This has given rise to a growing reliance on computational intelligence tools, particularly within
diverse communication networks, to predict bandwidth or congestion issues which can ultimately affect the quality of service (QoS) [3].

Utilizing the capabilities of artificial intelligence (AI) through neural networks, this work illustrates how AI can enhance QoS within a corporate network. By proactively predicting bandwidth and network resource requirements, we can represent the QoS for a communication network. The innate learning capability of a neural network presents it as a robust method for forecasting time series traffic.

In this context, it is imperative to provide a succinct overview of the types of neural networks employed in this study, in order to create intelligent decision-making agents for traffic bandwidth prediction:

- **Convolutional Neural Network (CNN):** CNNs consist of input, hidden, and output layers, with hidden layers encompassing convolution, normalization, pooling, and other layers. Originally developed for image processing by Hinton et al. (2006), CNNs have since found applications in network traffic analysis. CNNs’ fundamental components, filters or kernels, perform convolution operations to extract relevant features from input samples.

- **Recurrent Neural Network (RNN):** RNNs, due to their capacity to predict future events based on historical observations, inherently contain a form of memory. Their nodes are interconnected through loops. However, they often grapple with the ‘vanishing gradient’ problem, which limits the retention of data over a finite number of time steps. Long Short-Term Memory (LSTM) networks, a variant of RNNs designed to handle large-scale data, can mitigate this issue. Initially proposed by Hochreiter and Schmidhuber [68], LSTM networks consist of cells that use gates to add, remove, or modify data. These gates are composed of sigmoid and multiplication functions. Different versions of LSTM networks share common features, including three types of gates: forget, memory, and output gates.
5.3 Design Principles for Intelligent QoS Agents in Network Management

The theoretical foundation for employing an intelligent agent in network management, which was initially put forth in [150], is further expounded upon in the third chapter of this thesis. This agent boasts an additional layer of complexity through its capacity to exchange and process information from other agents as part of its decision-making protocol. In practice, an agent designed to make QoS decisions could consult with traffic prediction and traffic classification agents to ascertain the expected data flow through a specific link in the hours to come, and to identify incoming traffic types that possess specific QoS requirements [157]. The QoS agent could then leverage these forecasts and categorizations to guide its decision-making process in determining the ideal route for incoming services. This ensures that the defined level of QoS is provided to individual users, ultimately leading to an improved user experience.

Figure 5.1: An undirected graph representation of a network topology, illustrating the connectivity between various nodes.

The QoS agent incorporates measurements from the network state, including metrics such as link latency, jitter, traffic predictions, and traffic types, in combination with
prior knowledge, to devise the optimal network path and configuration. For instance, the process of determining the available routes between a pair of nodes within the network is explained using a simple graph shown in Figure 5.1.

In the realm of network design and analysis, the application of graph theory has emerged as an indispensable tool, offering powerful mathematical models to represent complex systems such as a campus network [45]. The undirected graph, in particular, forms a quintessential representation of a campus network, as it captures the essential features of such networks, including nodes (representing routers, servers, or switches) and edges (representing physical or wireless connections).

The undirected graph can be formally represented as $G = (V, E)$, where $V$ is the set of vertices (or nodes) and $E$ is the set of edges. An edge in an undirected graph is an unordered pair $u, v$ where $u, v \in V$ [26].

The selection of an undirected graph to represent a campus network arises from the following key characteristics of such networks:

- **Symmetry of Connections**: Campus networks typically exhibit bidirectional communication, which is effectively captured by an undirected graph. If a node $u$ is connected to node $v$, the communication can proceed in both directions, from $u$ to $v$ and vice versa. This is represented in an undirected graph as an edge $u, v$ that does not distinguish between $u$ to $v$ and $v$ to $u$.

- **Connectivity and Reachability**: An undirected graph allows the investigation of network connectivity, i.e., to verify whether every pair of nodes in the network is connected directly or indirectly. It also enables the analysis of reachability from any given node, which is pivotal in ensuring the robustness of the network [32].

- **Simplicity and Clarity**: The representation of a campus network as an undirected graph simplifies the visualization and comprehension of the network. Without directionality associated with the edges, an undirected graph provides a more intu-
itive and uncluttered model for non-hierarchical networks, making it particularly suitable for campus networks.

A more formal justification of the use of undirected graphs can be given through the concept of a "connected" graph. A graph is said to be connected if there exists a path between every pair of vertices. This is an essential condition for the effective functioning of a campus network as it guarantees the possibility of communication between any pair of nodes. Formally, a graph $G = (V, E)$ is connected if, for every pair of vertices $u, v \in V$, there exists a path from $u$ to $v$. Campus networks, to ensure maximum availability and reliability, are typically designed as connected networks, which can be efficiently modeled and analyzed using connected undirected graphs.

Thus, undirected graphs offer a robust mathematical model to capture the architectural nuances of campus networks, providing a foundation for advanced analysis such as shortest path determination, network flow optimization, and fault diagnosis, among others [44].

For the initial phase of our study, we investigated a campus network as a use case to simulate and collect network state measurements. These measurements would later serve as training data for both the Traffic Prediction and Classification components and the QoS agent’s decision-making facet. A campus network is symbolically depicted as an undirected graph. An undirected graph constitutes a series of nodes and edges, with the edges enabling bidirectional connections between nodes. In common graphical representations, these edges are portrayed as lines linking two nodes.

In our exploration of network dynamics, it becomes apparent that there may be a multitude of routes bridging two points of communication. We visualize a standard campus network model, wherein each unit, or node, $(n \in N)$ possesses the ability to transport packets based on predetermined flow protocols.

Let’s denote $G = (V, L)$ as an undirected graph, with:

- $V$ representing the suite of nodes, each endowed with unique processing and memory
attributes;

• $L$ referring to the array of connection-level entities, wherein every connection is equipped with a bandwidth capacity $B_{ij}$.

The totality of accessible routes from an arbitrary node $G_i$ to any other node is evaluated as follows:

$$P_{total} = \sum_{i=1}^{i} V_i b$$

(5.1)

In this equation, $P$ symbolises the count of accessible routes for a specified node. $V_i$ pertains to the arbitrary node being assessed, while $b$ is a binary determinant of the existence of a connection (edge) from the assessed node $V_i$ to any other node. A connection’s presence implies $b = 1$, while its absence implies $b = 0$. Each connection is designated a unique identifier to prevent reiteration during the agent’s route calculation for a given path.

For every connection interlinking nodes, the agent evaluates the Available Bandwidth (AVB) based on traffic prediction values. In this situation, the agent evades lengthier routes by factoring in latency and hop count. The AVB illustrates the traffic volume in the network at a specific time and the potential data volume that can be integrated into the network. It is a representation of the average unused capacity over a considered time interval.

The Available Bandwidth in the network can be projected using statistics derived from the controller. The throughput for all connections, as elaborated in [69] and [139], can be computed by:

$$Th_{ij} = (TB(t) - TB(t - T))/T$$

(5.2)

In this equation, $TB$ denotes the Data transmitted at a specified port, and $T$ is the sampling time. The real-time throughput, symbolised as $Th_{ins}$, of the connections
evolves over time. Conversely, the total capacity of the network is constant, as indicated
by the connection value $E_{ij}$. Therefore, the instantaneous AVB on a specific connection
is computed as:

$$AVB = C_{ij} - Th_{ij}$$  \hfill (5.3)

To calculate the AVB for an end-to-end route, the lowest AVB along the path must
be determined and employed in the path ranking procedure. This is owing to the fact
that, even if the majority of connections along a given end-to-end route possess more
bandwidth, the determining factor for route selection is the minimal AVB among the
interconnecting links, articulated as:

$$AVB_{i_{select}} = \min(C_{ij} - Th_{ij})$$  \hfill (5.4)

This link is taken into account by the QoS agent during its end-to-end path rank-
ing decision. Typically, the QoS agent computes the total available paths’ bandwidth,
$AVB_{pathBW}$, between origin and destination nodes, employing the predicted throughput,
$Th_{pred}$.

$$AVB_{pathBW} = AC - ATh_{pred}$$  \hfill (5.5)

Here, $A$ represents the adjacency matrix. The QoS agent ranks the available routes
based on the resultant AVB values, anticipated propagation latency, transmission delays,
and jitter. The QoS agent might also consult the traffic classifier agent [135] to identify
a specific traffic class, ensuring that the suitable route, $K_{path}$, is chosen for users’ traffic
transmission.

$$K_{path} = \alpha_{max}(AVB_{pathBW}), \beta_{min}(D), \gamma_{min}(J)$$  \hfill (5.6)
In this equation, $AVB_{\text{path BW}}$ symbolises the available bandwidth of a specific route, $J$ denotes the jitter experienced along the route, and $D$ signifies the route delay. The coefficients $\alpha$, $\beta$, and $\gamma$ determine the weight attributed to each parameter, contingent on the precedence given to these parameters in the QoS agent’s decision-making process. Subsequently, a configuration file is generated for the specific source-destination traffic flow. This file is then programmed into the forwarding switch, enabling the selection of the optimal route for a specific user.

5.3.1 Assessing the Impact of Intelligent QoS Agents on Network Efficiency and User Experience

We have constructed a model of an academic institution’s digital communication network, supported by the thorough collection and analysis of historical user behavior data. This unique data set provides insights into network traffic patterns, which are crucial for anticipating potential bottlenecks in network routes. By predicting these congestion points, we are able to identify alternate pathways and reevaluate their potential capacity by examining the feasibility of various detours.

Our innovative approach combines the predictive power of Deep Learning (DL) methodologies with network link metrics to significantly enhance the system’s overall effectiveness [58]. To validate our work, we juxtapose our findings with the well-established Open Shortest Path First (OSPF) routing protocol.

To assess the performance of our novel system, we apply the OSPF routing mechanism, maintaining the widely accepted assumption that the strength of a link is inversely proportional to its assigned weight, and traffic load is equitably divided among several pathways according to established norms [108]. The weight assignment in OSPF can range from 1 to 65,535, with lower bandwidth links given a higher weightage and vice versa. For instance, a network manager may allocate a weight of 20,000 to a low-bandwidth link and 20 to a link with extraordinarily high bandwidth. For comparison,
in a conventional OSPF system, we assigned a weight of 100 to aggregated links and 200
to links directly connected to terminal devices.

In this scenario, OSPF heavily depends on the number of hops when determining
the most expensive routes. To track and scrutinize the varying metrics of network per-
formance, a packet tracer tool is used. This instrument examines the diverse metrics
for different application Protocol Data Units (PDUs), including FTP, Database, Email,
Voice, among others.

Figure 5.2: Network Integration with SDN and OSPF: This figure depicts a NetSim-
created OSPF enterprise network integrated with SDN. It showcases the network topol-
yogy, nodes, and routing mechanisms.

Drawing on the traditional Open Shortest Path First (OSPF) routing approach, Fig-
ure 5.2 visibly demonstrates the potential for certain links to experience substantial traffic
while others remain underutilized. During a particular snapshot of network activity, links
26 and 41 may emerge as the most heavily loaded. However, regardless of how congested
a specific link may be, devices stationed at Branch 1 persistently choose the same 39–26
pathway to communicate with the headquarters (HQ). This unvarying routing decision
results in substandard Quality of Service (QoS) and Quality of Experience (QoE), manifested as increased latency and diminished data transfer rate, as depicted in Figure 5.2.

OSPF’s inherent load balancing feature can be harnessed when HQ needs to communicate with the Data Center, dispatching packets across two pathways (28-40-42 and 39-26-41). This method may temporarily alleviate congestion at a given snapshot. However, it may also undermine QoE and overall network efficacy due to the unpredictable jitter resulting from packets traversing two paths with noticeable delay differences. As demonstrated in Figure 5.3, certain applications can experience jitter that breaches the 30ms threshold, widely accepted as the limit for acceptable performance.

Figure 5.3: Comparison of DL-Based-SDN and OSPF with respect to Delay and Jitter: The figure provides a comparative analysis of DL-Based-SDN and OSPF, focusing on metrics such as delay and jitter.

An effective resolution to this quandary could involve a comprehensive understanding of the status of each network link within the system, empowering it to adaptively alter the routing pathway before traffic congestion materializes. This notion forms the foundation of the system we propose. By melding Artificial Intelligence (AI) and Software-Defined
Networking (SDN), the Quality of Service (QoS) agent gains the capacity to observe the condition of network links and the dynamics of the network, equipping it with the ability to adapt its strategy and response as needed.

Upon projecting the forthcoming statuses of the links (refer to Table 5.1), the SDN component modifies system parameters such as link cost and priority to suit the evolving network landscape. This results in the identification of the most favorable pathway for routing various forms of network traffic. In addition, the agent is designed to retain and rank several optimal alternative routes. This capability enhances the robustness and resilience of our network by reducing the time required for network convergence following link outages. As elucidated in the previous section, the adjustment of metrics, the ranking process, the calculation, and the selection of optimal routes are executed in accordance with the agent’s predefined mapping and ranking procedures.

Table 5.1. LSTM Predicted Throughputs for Links at One-Hour Intervals

<table>
<thead>
<tr>
<th>1 h Pred.</th>
<th>link-26</th>
<th>link-28</th>
<th>link-29</th>
<th>link-27</th>
<th>link-39</th>
<th>link-40</th>
<th>link-41</th>
<th>link-42</th>
<th>link-44</th>
<th>link-46</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>17.3</td>
<td>26.1</td>
<td>5.3</td>
<td>34.2</td>
<td>42.7</td>
<td>33.7</td>
<td>66.8</td>
<td>29</td>
<td>2.9</td>
<td>6.8</td>
</tr>
<tr>
<td>2</td>
<td>17.8</td>
<td>29.3</td>
<td>15.6</td>
<td>23</td>
<td>36.5</td>
<td>29.2</td>
<td>82.2</td>
<td>33.5</td>
<td>3.3</td>
<td>5.8</td>
</tr>
<tr>
<td>3</td>
<td>21.3</td>
<td>14.6</td>
<td>7</td>
<td>26.8</td>
<td>39.2</td>
<td>15.4</td>
<td>84.9</td>
<td>34</td>
<td>3.2</td>
<td>5.2</td>
</tr>
<tr>
<td>4</td>
<td>20.4</td>
<td>35.7</td>
<td>4.7</td>
<td>31</td>
<td>36.3</td>
<td>33.6</td>
<td>127.8</td>
<td>35.4</td>
<td>2.4</td>
<td>5.3</td>
</tr>
<tr>
<td>5</td>
<td>21.3</td>
<td>32.1</td>
<td>4.9</td>
<td>39.1</td>
<td>39.3</td>
<td>31.1</td>
<td>127.2</td>
<td>32.7</td>
<td>2.6</td>
<td>5.6</td>
</tr>
<tr>
<td>6</td>
<td>19.5</td>
<td>39</td>
<td>5.6</td>
<td>20.9</td>
<td>257.6</td>
<td>26.9</td>
<td>80.2</td>
<td>34</td>
<td>2.3</td>
<td>5.5</td>
</tr>
</tbody>
</table>

Throughout the experimental phase, our agent capably constructed an optimal pathway utilizing the 40, 46, 27, 29 links, which were anticipated to be underused during the subsequent operational window. In the face of congested links, the system excelled by circumventing these potential chokepoints, thus contributing to an improved Quality of Experience (QoE) and Quality of Service (QoS).

When benchmarked against the traditional Open Shortest Path First (OSPF) protocol, our advanced system displayed superior performance with regard to latency and jitter metrics, as illustrated in Figure 5.3. Harnessing the Software-Defined Networking
(SDN) framework as its primary control mechanism, our agent is proficiently prepared to handle any potential divergence between the forecasted operations and real-time network activities.

5.4 Augmenting Wireless Network Performance: A Comprehensive Exploration of Opportunistic Packet Scheduling Algorithms

Within the rapidly evolving milieu of wireless networks, strategic orchestration of packet scheduling materializes as a pivotal element shaping a myriad of system performance metrics, including throughput, fairness, and user Quality of Service (QoS). This doctoral research primarily hinges on a comprehensive investigation of salient packet scheduling methodologies pertinent to mono-cell, single-channel wireless frameworks. One crucial point to acknowledge in the context of the study on "Multi-Agent Systems: An Implementation of Autonomous Network System" is the intricacy of cellular organization within these networks. Specifically, individual cell management does not implicitly denote a restriction to single-cell architecture. This essentially communicates that the orchestration of each cell’s tasks can be independently managed, without any necessary interaction with the scheduling processes of other cells.

Such an organizational methodology is usually enabled by allocating unique frequency channels to individual cells. This effectively mitigates any major interference from neighboring cells or so-called co-channel cells, which operate on the same frequency bandwidth. These unique frequency assignments ensure that co-channel cells are positioned at a distance substantial enough to prevent any notable co-channel disruptions.

The allocation of these frequencies can be carried out in a static manner during the initial planning stage of the network. However, an alternative approach is to make this assignment dynamically in the operational stage, employing intelligent algorithms to monitor and manage interference levels.

Considering a single-channel per cell scenario, the scheduling process becomes con-
siderably straightforward, given that all users within a particular cell are operating on a shared channel, thereby time-sharing a singular frequency. This eliminates the complexities often associated with multi-channel scheduling, promoting efficient network operations.

The research extends a detailed examination of opportunistic schedulers, sophisticated algorithms that harness the inherent diversity across multiple users to bolster system throughput. This strategic augmentation inevitably fine-tunes the network’s overall operational efficacy.

Special emphasis is placed on resource allocation algorithms such as the Proportional Fair Channel Aware scheduler (PF) and its more comprehensive version, the Generalized Proportional Fair Channel Aware scheduler (GPF). These algorithms, designed for wireless communication systems, prioritize users based on superior channel quality and elevated buffer occupancy. Notably, the GPF extends an additional degree of flexibility with the incorporation of an alpha parameter, facilitating more nuanced priority calculations [131].

The zenith of the dissertation is marked with the introduction of a sophisticated scheduler iteration, the Adaptive Proportional Fair Channel Aware scheduler (AGPF). This scheduler, an extension of the GPF, integrates two auxiliary parameters enabling a dynamic equilibrium between peak throughput and optimal user fairness.

A rigorous comparative evaluation of this novel scheduler against its traditional predecessors exhibits notable results, demonstrating its potential to revolutionize the field of wireless communications. The promising outcomes of this study substantiate its invaluable contribution to the realm of network packet scheduling and wireless communication research.
5.4.1 Unveiling the Proportional Fair Scheduler: A Key Actor in Autonomous Network Systems

In the realm of multi-agent systems, particularly within the context of Autonomous Network Systems, efficient resource management is of paramount importance. One such strategy, which has been demonstrated to strike an effective balance between fairness and throughput, is the Proportional Fair Scheduler (PF Scheduler).

The PF Scheduler, (refer to Algorithm 8), operates on the fundamental principle of optimizing the product of user satisfaction and throughput. User satisfaction is a function of the current data rate as compared to the average data rate for each user, thus taking into account both the quality of service (QoS) perceived by the user and the overall network throughput. This has the effect of creating an equitable distribution of resources, ensuring that all users are provided with a fair level of service, while still leveraging the highest data rate achievable in fluctuating wireless channel conditions.

The elegance of the PF Scheduler lies in its strategic resource allocation, making it particularly suited for multi-agent environments where collaborative actions and decisions need to be made. By taking a snapshot of each user’s data rate, the scheduler can gauge the network’s capacity and divide resources in a manner that optimizes fairness and throughput simultaneously. This allows for a sophisticated balancing act, preventing situations where the rich get richer - a scenario where users with initially high data rates receive a larger share of the resources, thereby exacerbating inequalities.

Notably, the Proportional Fair Scheduler has proven highly successful in applications pertaining to autonomous network systems, notably in the facilitation of network resource management, coordination among diverse agents, and the dynamic adaptation to ever-changing network conditions. As such, the implementation of the Proportional Fair Scheduler can dramatically improve the efficiency and adaptability of multi-agent systems, which forms the crux of this dissertation.

This algorithm operates by taking into consideration both the current channel con-
Algorithm 8 Proportional Fair Scheduler

Require: Users $U = \{u_1, u_2, ..., u_n\}$, Channel Condition $C = \{c_1, c_2, ..., c_n\}$, Buffer Occupancy $B = \{b_1, b_2, ..., b_n\}$

Ensure: Scheduled Users $S = \{s_1, s_2, ..., s_n\}$

1: Begin
2: Initialize User Rates $R = \{0, 0, ..., 0\}$, Past Average Rates $A = \{0, 0, ..., 0\}$
3: for each time-slot $t$ do
4: for each user $u_i$ in $U$ do
5: Calculate current rate $r_i(t) = c_i(t) \times b_i(t)$
6: Update $R[i] = r_i(t)$
7: end for
8: Determine user $u_{max} = \arg\max_i \left( \frac{R[i]}{\text{max}(A[i], \epsilon)} \right)$ for $u_i$ in $U$ // $\epsilon$ is a small positive number to avoid division by zero
9: Update $S[t] = u_{max}$
10: Update $A[u_{max}] = (1 - \frac{1}{\tau}) \times A[u_{max}] + \frac{1}{\tau} \times R[u_{max}]$ // $\tau$ is a time window for averaging
11: end for
12: End

ditions and the buffer occupancy of each user at each time slot. The user with the maximum ratio of the current rate to the past average rate (thus ensuring proportional fairness) is then chosen for transmission. The past average rate is updated using an exponential window to provide more weight to recent rates, reflecting the time-varying nature of wireless channels. By continually adjusting the past average rates, the algorithm dynamically adapts to changing network conditions, thereby promoting fairness and optimizing throughput.

5.4.2 Generalized Proportional Fair Scheduler: A Fine-tuned Approach for Autonomous Network Systems

This work extends the discussion to the Generalized Proportional Fair Scheduler (GPF), a noteworthy evolution of the original Proportional Fair Scheduler, particularly relevant to the scope of autonomous network systems within the realm of Multi-Agent Systems. It is essential to note that the GPF inherits the PF’s objective of pursuing a balance between system throughput and fairness among users, thus ensuring a homogeneous distribution of
network resources. Yet, the GPF introduces an intriguing novelty through an additional parameter: alpha. The alpha parameter offers the system a more extensive latitude to modulate priorities in resource allocation [124].

The GPF scheduler, as depicted in Algorithm 9, integrates a sense of adaptability, capable of responding to diverse channel conditions by tuning the alpha parameter. Such an advanced feature significantly influences the decision-making process in a multi-agent network system, fostering an environment that allows agents to interact and learn from the continually shifting network conditions.

This enhancement is particularly crucial in autonomous network systems, as it not only increases the degree of adaptability and responsiveness to the dynamism inherent to such environments, but it also empowers the system with a capability to tailor the trade-off between network throughput and fairness in resource allocation. By doing so, the GPF scheduler aligns perfectly with the key objective of autonomous network systems: achieving the highest possible level of efficiency and performance while ensuring fairness among all agents in the system.

In this vein, the introduction of the alpha parameter in the GPF scheduler can be interpreted as the catalyst to introduce a layer of machine learning. The ML algorithms can be employed to learn from network conditions and adaptively set the alpha value, thereby optimizing resource allocation in real-time. The resultant effect is a profound boost to the performance of autonomous network systems, underlining the utility of the GPF scheduler in the ambit of multi-agent systems.

The actual value for \texttt{alpha} isn’t given, as it is a customizable parameter that allows the system designer to fine-tune the balance between throughput maximization and fairness. Similarly, actual function implementations of the rate calculations are not provided, as they may vary depending on the specific system setup and channel conditions.
Algorithm 9 Generalized Proportional Fair Scheduler

Require: Users $U = \{u_1, u_2, ..., u_n\}$, Channel Condition $C = \{c_1, c_2, ..., c_n\}$, Buffer Occupancy $B = \{b_1, b_2, ..., b_n\}$

Ensure: Scheduled Users $S = \{s_1, s_2, ..., s_n\}$

1: Begin
2: Initialize User Rates $R = \{0, 0, ..., 0\}$, Past Average Rates $A = \{0, 0, ..., 0\}$, Alpha $\alpha$
3: for each time-slot $t$ do
4: for each user $u_i$ in $U$ do
5: Calculate current rate $r_i(t) = c_i(t) \times b_i(t)$
6: Update $R[i] = r_i(t)$
7: Update $B[i] = b_i(t)$
8: end for
9: Determine user $u_{max} = \arg \max_i \left( \frac{R[i]^\alpha}{\max(A[i], \epsilon)} \times B[i] \right)$ for $u_i$ in $U$ // $\epsilon$ is a small positive number to avoid division by zero
10: Update $S[t] = u_{max}$
11: Update $A[u_{max}] = (1 - \frac{1}{\tau}) \times A[u_{max}] + \frac{1}{\tau} \times R[u_{max}]$ // $\tau$ is a time window for averaging
12: end for
13: End

5.4.3 Adaptive Generalized Proportional Fair Scheduler

In the realm of contemporary wireless communication systems, the role of sophisticated scheduling algorithms is paramount in improving the balance between system throughput and fairness among users. Among these, the Adaptive Generalized Proportional Fair Scheduler (AGPF) stands as an innovative development that modifies the functionality of the Generalized Proportional Fair Scheduler (GPF). It introduces the capability to adapt to the dynamic nature of network conditions, which is a critical factor in the successful operation of autonomous multi-agent systems.

The pivotal innovation of the AGPF (refer to Algorithm 10) lies in its unique characteristic to adjust the $\alpha$ parameter, a crucial component in determining the priority calculation within the GPF scheduling scheme. This adaptive $\alpha$ parameter is a dynamic entity that evolves at each time step, being influenced by the prevailing buffer size of each user within the network. The adjustments are performed in such a manner that when the buffer size of any user exceeds a pre-defined threshold, the $\alpha$ param-
eter is reduced to a minimum value ($alpha_{min}$). Conversely, when the buffer size is within the acceptable threshold, the alpha parameter inclines towards a maximum value ($alpha_{max}$).

**Algorithm 10 Adaptive Generalized Proportional Fair Scheduler (AGPF)**

**Require:** Users $U = \{u_1, u_2, ..., u_n\}$, Channel Condition $C = \{c_1, c_2, ..., c_n\}$, Buffer Occupancy $B = \{b_1, b_2, ..., b_n\}$, $\alpha_{max}$, $\alpha_{min}$, Buffer Size Threshold $\theta$

**Ensure:** Scheduled Users $S = \{s_1, s_2, ..., s_n\}$

1: Initialize User Rates $R = \{0, 0, ..., 0\}$, Past Average Rates $A = \{0, 0, ..., 0\}$

2: Initialize $\alpha = \alpha_{min}$

3: for each time-slot $t$ do

4: for each user $u_i$ in $U$ do

5: Calculate current rate $r_i(t) = c_i(t) \times b_i(t)$

6: Update $R[i] = r_i(t)$

7: end for

8: Calculate mean buffer size $B_{mean} = \frac{1}{n} \sum_{i=1}^{n} B[i]$

9: Update $\alpha = \alpha_{min} + (\alpha_{max} - \alpha_{min}) \times e^{-\theta \times B_{mean}}$

10: Determine user $u_{max} = \text{arg max}_i \left( \frac{R[i]}{(A[i] + o)^\alpha} \right)$ for $u_i$ in $U$

11: Update $S[t] = u_{max}$

12: Update $A[u_{max}] = (1 - \frac{1}{\tau}) \times A[u_{max}] + \frac{1}{\tau} \times R[u_{max}]$

end for

The priority calculations within the AGPF are executed as follows:

```python
self.alpha = self.alpha_min + (self.alpha_max - self.alpha_min) * np.exp(-self.buffer_size_threshold * buffer_size_per_ue.mean())
priorities = (1 + o) / (b ** self.alpha) * buffer_size_per_ue * se
```

This innovative alpha parameter adjustment mechanism allows the AGPF to optimize the trade-off between fairness and throughput based on the real-time network conditions. By incorporating this ability to adapt and evolve in response to changing conditions, the AGPF sets the stage for higher efficiency in resource allocation and network performance in the context of autonomous multi-agent systems.

Table 5.2 encapsulates an essential comparative perspective between the three distinct
schedulers – Proportional Fair Scheduler, Generalized Proportional Fair Scheduler, and the Adaptive Generalized Proportional Fair Scheduler. Exhibiting the most relevant aspects of each algorithm, the table serves as a comprehensive yet concise summary that enables readers to grasp the essential differences swiftly. This summary dissects how each scheduler ascertains priorities, their respective adaptivity features, consideration towards network conditions, and their optimization objectives.
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$R[i]/\max(A[i], \epsilon)$</td>
<td>$R[i]/(A[i] + o)^\alpha$</td>
<td>$R[i]/(A[i] + o)^\alpha$ where $\alpha$ is adaptive</td>
</tr>
</tbody>
</table>

| Adaptivity           | Fixed algorithm parameters      | Fixed $\alpha$ parameter                | $\alpha$ parameter adjusts adaptively              |

| Consideration of Network Conditions | Not directly considered | Not directly considered | Uses average buffer size to adapt $\alpha$ parameter |

| Optimization Objectives | Balances fairness and throughput | Flexibly balances fairness and throughput with $\alpha$ parameter | Dynamically optimizes balance between fairness and throughput based on network conditions |

Table 5.2. Key differences among PF, GPF, and AGPF opportunistic Schedulers.
5.4.4 Empirical Exploration: Delving into Scheduler Dynamics

As we navigate into the empirical realm of this dissertation, we are set to elucidate the functional dynamics of our three central schedulers – the Proportional Fair Scheduler, the Generalized Proportional Fair Scheduler, and the Adaptive Generalized Proportional Fair Scheduler. This experimental section aims to shed light on the intricate operation of these schedulers under diverse network conditions, thereby providing tangible evidence of their theoretical properties and potential real-world implications. Guided by rigorous methodologies and comprehensive evaluation metrics, we intend to substantiate our theoretical propositions and contribute meaningfully to the broader discourse surrounding the optimization of autonomous network systems. Let us now embark on this journey of empirical discovery.

![Line graph showcasing performance trend for each scheduling scheme.](image1)
![Bar chart comparing rewards obtained for varying alpha configurations.](image2)

Figure 5.4: Performance comparison of PF, GPF, and AGPF scheduling schemes across different NPRBS values.

The experimentation results of the different scheduling algorithms presented are indicative of the impact of network resource allocation strategies on overall system performance, as depicted in Table 5.3 and Figure 5.4. Further details pertaining to the implementation process are thoroughly discussed and presented in Appendix C for reader
<table>
<thead>
<tr>
<th>nprbs</th>
<th>Scheme</th>
<th>$\alpha$</th>
<th>$\alpha_{\text{min}}$</th>
<th>$\alpha_{\text{max}}$</th>
<th>Reward</th>
<th>Runtime</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>PF</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-2208.64</td>
<td>0:09:34</td>
</tr>
<tr>
<td></td>
<td>GPF</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>-1798.08</td>
<td>0:10:40</td>
</tr>
<tr>
<td></td>
<td>AGPF</td>
<td>0</td>
<td>0.001</td>
<td>0.1</td>
<td>-1798.57</td>
<td>0:10:43</td>
</tr>
<tr>
<td></td>
<td>AGPF</td>
<td>0</td>
<td>0.1</td>
<td>0.1</td>
<td>-1836.74</td>
<td>0:10:25</td>
</tr>
<tr>
<td></td>
<td>AGPF</td>
<td>0.9</td>
<td>0.1</td>
<td>0.8</td>
<td>-1836.74</td>
<td>0:09:23</td>
</tr>
<tr>
<td></td>
<td>AGPF</td>
<td>0.5</td>
<td>0.1</td>
<td>0.5</td>
<td>-1836.74</td>
<td>0:10:29</td>
</tr>
<tr>
<td></td>
<td>AGPF</td>
<td>0.1</td>
<td>0.1</td>
<td>0.9</td>
<td>-1836.74</td>
<td>0:10:13</td>
</tr>
<tr>
<td></td>
<td>AGPF</td>
<td>0</td>
<td>0.00001</td>
<td>0.00009</td>
<td>-1798.11</td>
<td>0:11:42</td>
</tr>
<tr>
<td>25</td>
<td>PF</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-1761.02</td>
<td>0:09:18</td>
</tr>
<tr>
<td></td>
<td>GPF</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>-1443.04</td>
<td>0:08:54</td>
</tr>
<tr>
<td></td>
<td>AGPF</td>
<td>0</td>
<td>0.001</td>
<td>0.1</td>
<td>-1443.48</td>
<td>0:10:02</td>
</tr>
<tr>
<td></td>
<td>AGPF</td>
<td>0</td>
<td>0.1</td>
<td>0.1</td>
<td>-1473.11</td>
<td>0:10:43</td>
</tr>
<tr>
<td></td>
<td>AGPF</td>
<td>0.9</td>
<td>0.1</td>
<td>0.8</td>
<td>-1473.12</td>
<td>0:10:26</td>
</tr>
<tr>
<td></td>
<td>AGPF</td>
<td>0.5</td>
<td>0.1</td>
<td>0.5</td>
<td>-1473.11</td>
<td>0:11:13</td>
</tr>
<tr>
<td></td>
<td>AGPF</td>
<td>0.1</td>
<td>0.1</td>
<td>0.9</td>
<td>-1473.11</td>
<td>0:10:53</td>
</tr>
<tr>
<td></td>
<td>AGPF</td>
<td>0</td>
<td>0.00001</td>
<td>0.00009</td>
<td>-1443.05</td>
<td>0:09:28</td>
</tr>
<tr>
<td>30</td>
<td>PF</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-1464.61</td>
<td>0:09:19</td>
</tr>
<tr>
<td></td>
<td>GPF</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>-1205.62</td>
<td>0:10:26</td>
</tr>
<tr>
<td></td>
<td>AGPF</td>
<td>0</td>
<td>0.001</td>
<td>0.1</td>
<td>-1205.88</td>
<td>0:10:02</td>
</tr>
<tr>
<td></td>
<td>AGPF</td>
<td>0</td>
<td>0.1</td>
<td>0.1</td>
<td>-1230.80</td>
<td>0:09:44</td>
</tr>
<tr>
<td></td>
<td>AGPF</td>
<td>0.9</td>
<td>0.1</td>
<td>0.8</td>
<td>-1230.80</td>
<td>0:09:14</td>
</tr>
<tr>
<td></td>
<td>AGPF</td>
<td>0.5</td>
<td>0.1</td>
<td>0.5</td>
<td>-1230.80</td>
<td>0:10:24</td>
</tr>
<tr>
<td></td>
<td>AGPF</td>
<td>0.1</td>
<td>0.1</td>
<td>0.9</td>
<td>-1230.80</td>
<td>0:08:56</td>
</tr>
<tr>
<td></td>
<td>AGPF</td>
<td>0</td>
<td>0.00001</td>
<td>0.00009</td>
<td>-1205.61</td>
<td>0:09:35</td>
</tr>
</tbody>
</table>
Through meticulous comparative analysis, we discern how the schedulers’ performance diverges across different parameter configurations and resource blocks (nprbs), specifically the alpha parameter governing the degree of fairness and throughput optimization in the scheduling process.

The Proportional Fair (PF) Scheduler operates with a fixed fairness approach, devoid of any alpha value, which is unique to its operation. This absence of alpha value differentiates it from other scheduling strategies and characterizes its specific performance. This scheduler prioritizes users based solely on their instantaneous channel conditions. The associated rewards show a systematic improvement as the number of Physical Resource Blocks (PRBs) - denoted as n_prbs - increases from 20 to 30, reflecting its tendency towards enhanced performance under more generous resource allocation conditions. However, despite these advancements, the PF Scheduler appears to be consistently outperformed by its more sophisticated counterparts, the Generalized Proportional Fair (GPF) and the Adaptive Generalized Proportional Fair (AGPF) schedulers.

The GPF Scheduler, with its flexible alpha parameter (varying from 0 to 1), offers a hybrid approach encompassing both proportional fairness and max-throughput schemes. This ability to strike a balance between system throughput and fairness culminates in superior performance compared to the PF Scheduler, as evidenced by the higher rewards across all levels of n_prbs. Notably, the GPF Scheduler’s alpha value of 0 implies a focus on optimizing system throughput, disregarding user fairness.

The AGPF Scheduler, as the most adaptive variant, exhibits a range of performances based on its alpha, alpha_min, and alpha_max parameters. When alpha_min and alpha_max are set to lower values, the AGPF Scheduler’s performance closely mirrors that of the GPF Scheduler with alpha equal to 0, alluding to its ability to successfully adapt to the prevailing network conditions. However, when alpha values and ranges are higher, the AGPF Scheduler incurs slightly lower rewards, indicating the impact of
maintaining a higher degree of user fairness on overall system performance.

It is noteworthy that the AGPF Scheduler’s completion times are relatively consistent across all test cases, underscoring its robustness in handling different system configurations. These completion times provide further evidence of the scheduler’s adaptability and its potential to ensure reliable system performance across varied network conditions.

In conclusion, the AGPF Scheduler’s capacity to dynamically adjust its parameters based on real-time network conditions positions it as a highly promising tool for optimizing autonomous network systems, subject to trade-offs between system throughput and user fairness. Future work could further finetune these parameters to optimize its performance under an expanded array of network scenarios.

5.5 Chapter Summary

The maiden segment of this chapter embarks on an intricate exploration, critically examining the fusion of Deep Learning paradigms with network link metrics, specifically targeting the enhancement of digital communication networks within an academic milieu. Relying upon a distinct dataset - a compendium of historical user behaviors - this chapter adeptly unveil the nuances of network traffic patterns and consequently identifies areas prone to bottlenecks, laying down a robust foundation for the innovative system proposed.

Employing the classic Open Shortest Path First (OSPF) routing protocol as a comparative benchmark, the analysis brought forth pivotal insights. Chiefly, it became evident that OSPF’s inherent reliance on hop counts fosters inequitable traffic distribution, with certain links being inundated whilst others languish underutilized. Such imbalances spawn diminished Quality of Service (QoS) and Quality of Experience (QoE) indices, manifested in exacerbated latency and curtailed data transfer velocities.

Emerging as an intellectual riposte to these challenges, the chapter heralds an avant-garde system, a symbiosis of Artificial Intelligence (AI) and Software-Defined Networking
Birthed from this union is a sagacious QoS agent, demonstrably proactive and replete with knowledge of the network’s intricate dynamics. Possessing the prowess to preemptively recalibrate routing vectors prior to congestion onset, this agent epitomizes efficiency. Its dexterity extends beyond mere route identification to curating a stratified hierarchy of optimal alternative conduits, imbuing the network with a commendable resilience.

Empirical evaluations illuminate the formidable capabilities of our proposed paradigm. Notably, the system’s architectural brilliance shines through in constructing ideal pathways, adeptly sidestepping prospective congestion nexuses, and manifestly elevating both QoS and QoE. When juxtaposed against the OSPF, our system’s performance is conspicuously superior, particularly regarding latency and jitter. The system’s astute preparedness to reconcile anticipated and real-time network dynamics, a direct upshot of the ingrained Software-Defined Networking (SDN) framework, further accentuates its merits.

Transitioning to the latter segment of the chapter, the narrative journeys through the intricate terrains of wireless communication, with a special emphasis on the art and science of packet scheduling in mono-cell, single-channel wireless systems. Methodological rigor, fused with empirical validations, forms the crucible of this section’s narrative.

- **The Autonomy of Cellular Organization:** Delving deep into the architectural DNA of cellular networks, the exposition underscores cells’ intrinsic autonomy, an accomplishment largely attributable to refined frequency allocations, dynamically sculpted by sagacious algorithms.

- **Streamlining through Single-Channel Per Cell:** By canonizing a uniform frequency for users within a cell, the narrative expounds on the strategic bypass of multi-channel scheduling complexities, heralding an era of amplified operational efficacy.

- **Emergence of Opportunistic Schedulers:** The discourse pivots to the advent
of groundbreaking opportunistic schedulers, like the PF and GPF. Their acute sensitivity to diverse user demographics, fluctuating channel conditions, and buffer occupancy have indisputably augmented system bandwidth. GPF, with its ingenious parameter, offers a granular touch to resource distribution.

- **AGPF – The Pinnacle of Scheduling Algorithms:** The AGPF Scheduler, seamlessly melding peak bandwidth with egalitarian user access, emerges as the zenith in scheduling innovation. It stands as a testament to the untapped potential of adaptive protocols in wireless realms.

- **Empirical Exploration of Schedulers:** A scrupulous, data-centric approach demystifies the operational trajectories of the PF, GPF, and AGPF schedulers. PF’s amplified efficacy under escalated PRBs, though commendable, is overshadowed by GPF’s adaptability, bolstered by its parameter. AGPF’s chameleon-like adaptiveness, evidenced by stable completion timelines across diverse configurations, underscores its unparalleled supremacy.

In its entirety, the chapter unravels the enigmatic tapestry of avant-garde packet scheduling algorithms, simultaneously lauding their transformative potential. These pioneering methodologies, exuding adaptability, fairness, and heightened performance, are poised to redefine wireless communication’s very lexicon. In its concluding stride, the chapter casts an aspirational gaze, beckoning subsequent scholarly pursuits that further amplify these algorithms’ efficacy in an ever-diversifying network milieu, all within the broader context of Multi-Agent Systems and Autonomous Network Systems.
Chapter 6


This chapter delves into the innovative incorporation of Artificial Intelligence/Machine Learning (AI/ML) as crucial agents within multi-agent systems aimed at managing and optimizing the performance of 6G networks. It underscores the transformative shift towards these AI/ML agents, primarily focusing on the novel Speed-optimized LSTM (SP-LSTM) model and Reinforcement Learning. The chapter delineates the unique roles these agents play within the system, their interaction dynamics, and the resultant impacts on proactive management and dynamic routing in 6G networks. Furthermore, it explores the potential of such AI/ML-driven multi-agent systems in reshaping the future of network management, catering to 6G’s demanding requirements of ultra-low latency, ultra-reliability, and efficient heterogeneity management. Ultimately, this chapter forms a bridge between AI/ML advancements and autonomous network systems, laying the groundwork for the transformative role of multi-agent systems in future network management.

6.1 Introduction

As 6G technology unfolds, it is bringing forth an era of accelerated connectivity, exceptional capacity, and an array of diverse devices that holds the potential to redefine our digital experience [25], [130], [118]. However, the progressive nature of these attributes presents unprecedented challenges for efficient network management and optimization [130]. Legacy network management strategies, anchored in rigid and predefined routing rules, are ill-equipped to navigate the dynamic and complex 6G landscape [9], [126], [164].
These conventional techniques falter when confronted with the data deluge expected from 6G networks, thereby underutilizing resources and limiting network performance [170], [70].

One of the principal requirements of 6G networks is ultra-low latency, a critical factor for applications like autonomous vehicles, real-time gaming, and telesurgery [143]. Network congestion, a common issue in large-scale systems, often exacerbates latency problems, particularly when rerouting is necessary to manage network traffic [149]. To combat these challenges, we propose an AI/ML-based network management framework that proactively employs predictive analytics and dynamic routing.

Predictive analytics, in this context, forecasts future network congestion based on past network usage data, allowing network managers to pre-emptively balance network load and minimize latency [85]. Long Short-Term Memory (LSTM) networks, a subset of recurrent neural networks, have demonstrated impressive efficiency in capturing long-term dependencies in time-series data [68]. However, traditional LSTM models are computationally intensive, leading to slower training times and delayed responses to rapidly evolving network conditions. Moreover, their performance degrades with complex temporal dependencies over extended time lags [104], [121]. To overcome these limitations, we propose the Speed-optimized LSTM (SP-LSTM), a faster and more efficient alternative that maintains high predictive accuracy in line with the real-time requirements of 6G networks.

Dynamic routing, on the other hand, adjusts routing decisions according to the current network state [87]. Reinforcement Learning (RL), a subset of machine learning where an agent learns to maximize rewards through interaction with its environment, offers promising potential for dynamic routing [163]. By factoring in both the current and predicted future network states, the RL agent can make more informed routing decisions, optimizing network performance while maintaining low latency [99]. By utilizing Q-learning in networks, the RL agent can efficiently reduce the overhead for route search-
ing, enabling more informed routing decisions that optimize network performance while maintaining low latency.

In this chapter, we introduce a two-tiered AI/ML system that marries these two techniques. Through the combination of predictive analytics and dynamic routing, we aim to meet the ultra-low latency, high reliability, and heterogeneity management requirements of 6G networks. As 6G networks evolve, it’s essential that our AI/ML models learn and adapt, continuously updating their knowledge base on historical and real-time data [169].

The implementation of AI/ML into network management systems is paramount for harnessing the full potential of 6G networks. Our proposed solution constitutes a significant advancement, showing the potential of AI/ML to effectively manage the vast amount of data in 6G networks and make near-real-time routing decisions, thereby optimizing network performance.

6.1.1 Background and Need for Enhanced 6G Network Management

The dawning age of 6G networks presents a seismic alteration in network management’s topography, spurred by unparalleled data volumes, varied network devices, and demanding ultra-low latency prerequisites [25], [130]. These notable attributes drive the need for innovative management techniques as conventional ones prove insufficient for such a dynamic environment [2]. Recent research advancements have proposed AI/ML-based network management solutions that exhibit proactive characteristics, marking a departure from the reactive nature of traditional strategies and paving the way for more versatile and adaptive methods [23].

Traditionally, network management has been rooted in static, preordained routing protocols. This fixed strategy becomes markedly ineffective within the domain of 6G networks, underlining the urgency for flexible mechanisms adept at coping with the unprecedented scale and diversity of data. Moreover, traditional methodologies fail to leverage the vast data potential inherent in 6G networks, signaling the need for trailblaz-
ing approaches for optimizing resource utilization and network performance [167].

Within the context of evolving 6G networks, achieving ultra-low latency is paramount, specifically for applications that are latency-sensitive like autonomous driving, real-time gaming, and remote medical procedures. Despite advancements, the legacy issue of latency in dense network environments continues to persist, magnified particularly when network traffic is rerouted to reduce congestion. To navigate these hurdles, the scientific community is increasingly exploring the potency of AI/ML solutions, experimenting with network management frameworks that leverage predictive analytics and adaptive routing.

Predictive analytics stands out as an instrumental approach in this context, enabling network administrators to anticipate potential congestion scenarios and accordingly orchestrate preemptive measures. In this domain, Long Short-Term Memory (LSTM) networks, a distinct subclass of recurrent neural networks, have demonstrated significant competence due to their capacity to understand intricate, long-term correlations in time series data [57], [104].

At the same time, there is a rising interest in dynamic routing, a strategy that adapts routing decisions in line with real-time network conditions. Reinforcement Learning (RL), an AI/ML strategy where an agent learns to optimize a reward function through environmental interaction, has proven its mettle for adaptive routing applications [5]. By considering both the current network state and anticipated future scenarios, RL enables more informed routing decisions, optimizing network performance while maintaining low latency.

The relentless evolution of 6G networks accentuates the imperative for AI/ML models that learn and adapt continuously. The dynamism of these networks necessitates models that not only learn from historical and present data but are also capable of adjusting to future changes. The integration of AI/ML into network management systems is therefore crucial to fully exploit the potential of 6G networks.

The current state of research underscores the paradigm-shifting potential of AI/ML-
based techniques in governing the performance of 6G networks. It also emphasizes the urgent need for continued exploration and refinement to harness the full benefits of predictive analytics, dynamic routing, and continuous learning in the management and optimization of next-generation networks.

6.1.2 Role of Multi-Agent Systems in 6G Networks

The inception of 6G networks brings about a prominent shift in the realm of network management, requiring strategies that can match the complexity, variability, and ultra-low latency demands of these networks. The traditional network management methods, which are static and pre-defined, struggle to navigate this dynamically changing landscape, signaling an imperative for more adaptable and dynamic methods. In addressing this need, multi-agent systems (MAS), a key aspect of artificial intelligence, have emerged as a promising solution.

Multi-agent systems (MAS) denote systems comprised of multiple interacting intelligent agents, which can be utilized to solve problems that are difficult or impossible for a single agent or monolithic system to solve. In the context of 6G networks, these systems have the potential to revolutionize network management and optimization, given their capacity to autonomously monitor, adapt, and optimize network functions, without requiring human intervention.

The potential of MAS is particularly evident in addressing the latency challenge pervasive in 6G networks. By leveraging the decentralized nature of MAS, network managers can create systems that proactively detect and mitigate potential congestion points, thereby enhancing the speed and reliability of data transmission. Moreover, through the integration of machine learning techniques, these MAS can be trained to continuously learn and adapt to changes in the network environment, thereby promoting a more efficient utilization of network resources.

At the intersection of MAS and Reinforcement Learning (RL), dynamic routing can
be achieved with unprecedented efficiency. RL, an AI/ML technique in which an agent learns to optimize a reward function through interaction with its environment, demonstrates its viability for dynamic routing applications. Coupled with MAS, it facilitates the orchestration of intelligent, context-aware routing decisions, optimizing network performance while securing ultra-low latency.

The inherent capability of MAS to operate in a distributed and autonomous manner makes them an ideal solution for managing the vast and diverse array of devices that make up 6G networks. Through distributed intelligence, MAS can ensure the harmonious coexistence of multiple network devices, adapting to fluctuations in network traffic, and maintaining optimal performance.

As the 6G network continues to evolve, the need for solutions that can adapt to changes and learn from past experiences is more critical than ever. In this light, the potential of MAS to foster a continuous learning environment presents a path towards a more efficient and intelligent network management system, thereby maximizing the potential of 6G networks.

The deployment of MAS in 6G networks signals a transformative shift towards more intelligent and autonomous network management systems. However, despite the promising advancements, there is a pressing need for further exploration and refinement of these systems to fully leverage the benefits of MAS in the context of 6G network management.

In the evolving landscape of communications, the 6G network architecture is on the horizon, promising a transformative leap forward. While its definitive architecture remains under exploration by multiple organizations, our proposed model, as depicted in Figure 6.1, presents a harmonious convergence of groundbreaking elements.

In the architectural tapestry of the emerging 6G network, several key components come to the fore. BS represents the Base Station, while UE is the User Equipment, interfacing with Service Centers (SC) and Edge Clouds (EC). UDR stands as the User Data Repository, while the SM, MEC, and HetNet signify Surface Modem, Mobile Edge
Computing, and Heterogeneous Network respectively. On the underwater front, UWC
denotes Underwater Communication, augmented by Intrusion Nodes (IN), Energy Har-
vesters (EH), and Quantum Technology modules (QT). The aerial dimension houses SAT
(Satellite), Communication Clouds (CC), Network Cloud (NC), and Service Instances
(SI). Rising higher, we encounter HAPs (High-Altitude Platforms), Drones, and Ter-
ahertz modules (THz). The AI/ML landscape embeds Artificial Intelligence/Machine
Learning nodes, intertwined with Privacy Agents (PA) and Data Repositories (DR).
Supplementing these, SON denotes Self-Organizing Networks, LEO brings in Low Earth
Orbit satellites, RF encapsulates Radio Frequency, and V2X, LPWAN, and RAN depict
Vehicle-to-Everything, Low-Power Wide-Area Network, and Radio Access Network re-
spectively.

At its nexus is the User Equipment (UE), forging pivotal connections to the Base Sta-
tion (BS), Service Center (SC), and AI/ML modules, the latter introducing a paradigm
shift in intelligent operations. The UE’s versatility is underscored by its hybrid liaisons
with the Underwater Communication (UWC) and wireless associations with the User
Data Repository (UDR). The terrestrial foundation, exemplified by the BS and Edge
Cloud (EC), beautifully dovetails with the aerial dimension, where the Satellite (SAT)
bridges connections to High-Altitude Platforms (HAPs) and drones. Essential agents like
the Surface Modem (SM), Network Cloud (NC), and Quantum Technology (QT) weave
throughout, while entities like V2X and LPWAN broaden the architecture’s purview.
Furthermore, the LEO satellites, Radio Frequency modules, and other advanced tech-
nologies accentuate its depth and breadth. Each connection, delineated distinctly—solid
lines for landlines, dashed for wireless, and dotted for hybrid—encapsulates the intricate
interplay of diverse communication modalities. Amidst the ongoing discourse on 6G’s
ultimate form, our model underscores its potential to redefine connectivity, melding ver-
satility with intelligence.
Figure 6.1: The comprehensive 6G network architecture with AI/ML agents for network performance enhancement.
6.2 AI/ML-Agents and Their Paradigm Shift in Network Management

6.2.1 Overview of AI/ML-Agents in Network Management

With the surge of digital transformation, the telecommunications industry is experiencing unprecedented challenges and opportunities. The vast increase in the volume of data, the advent of new technologies such as 5G/6G, and the increasing demand for high-speed, reliable, and secure connections underscore the critical role of advanced network management strategies. To effectively handle these evolving requirements, the application of Artificial Intelligence (AI) and Machine Learning (ML) in network management emerges as a potent solution.

AI and ML have paved the way for autonomous network systems characterized by their capacity to self-manage, adapt, and optimize network performance under changing conditions. These technologies introduce intelligent agents capable of interacting with the environment, learning from experiences, making decisions, and executing actions to achieve specific goals [129].

In network management, AI/ML agents can contribute to various domains. They can help enhance network performance, predict and manage network traffic, detect and mitigate cyber threats, optimize resource allocation, and improve service quality, to name a few [34].

Incorporating AI/ML in multi-agent systems has opened up new horizons for autonomous network systems. Multi-agent systems involve several intelligent agents working together to solve complex tasks beyond individual agents’ capabilities [161]. These systems have found their way into the telecommunications industry, promising to revolutionize the way we manage networks. The internal architecture of the Intelligent Agent has been thoroughly outlined in Chapter 3.

The essence of this dissertation lies in exploring the implementation of AI/ML-powered multi-agent systems in autonomous network management. By delving into
various aspects of these advanced technologies and dissecting their applications, this dissertation aims to shed light on the potential and challenges of deploying multi-agent systems in the context of 6G networks.

6.2.2 Integrating AI/ML into BDI Model for Multi-Agent Systems

Intelligent agents operate based on an advanced computational model known as the Belief-Desire-Intention (BDI) model [125]. This model essentially captures an abstract representation of the cognitive architecture of these agents. In this model, the beliefs of an agent represent information or knowledge about the world. The agent’s desires represent the objectives or goals that it seeks to accomplish, while its intentions reflect the commitments or actions that the agent plans to perform in order to realize these goals [30].

Formally, the BDI model can be represented as a tuple $BDI = (B, D, I)$, where $B \subseteq \Omega$ represents the beliefs of the agent, $D \subseteq 2^\Omega$ signifies the desires, and $I \subseteq 2^\Omega$ corresponds to the intentions of the agent. Here, $\Omega$ is the set of all possible states of the world. It is worth noting that the beliefs, desires, and intentions are interrelated, often influencing each other [161]. For instance, an agent’s desires and intentions may be updated based on changes in its beliefs about the state of the world.

However, this standard BDI model can be extended to incorporate machine learning or perception, thus giving rise to more sophisticated and adaptive agents. For instance, one could consider an agent’s beliefs to be not merely static facts about the world, but predictive models that the agent has learned from its past experiences. In this case, the belief function could be seen as a mapping from past experiences to future predictions, i.e., $B : Histories \rightarrow Probabilities(\Omega)$, where Histories denotes the set of all possible sequences of states and actions, and $Probabilities(\Omega)$ represents the set of all probability distributions over $\Omega$ [129]. This function $B$ can be updated over time using machine learning algorithms as the agent amasses more experience.
Beliefs

Desires

Intentions

(a) Basic BDI (Beliefs-Desires-Intentions) Model

(b) Enhanced BDI Model Incorporating AI/ML Concepts

Perception can also be integrated into the agent’s beliefs. Here, $B$ could be conceived as a function from sensory inputs to states of the world: $B : SensoryInputs \rightarrow \Omega$. This function $B$ could be learned using supervised learning if the agent has access to labeled training data, or it could be learned using unsupervised or reinforcement learning if the agent must learn from unlabeled data or through interaction with the environment [142].

In essence, the extension of the BDI model to incorporate machine learning and perception, as depicted in Figure 6.2, is a significant advancement in the field of AI and multi-agent systems. It paves the way for the development of more autonomous, adaptable, and intelligent agents capable of learning from their experiences and effectively interacting with their environment.

The advent of Artificial Intelligence (AI) and Machine Learning (ML) heralds a significant paradigm shift within multi-agent systems, leading to a notable transformation in 6G network performance enhancement. Traditionally, the Belief-Desire-Intention (BDI) model was considered a blueprint for modeling intelligent agents.
Presently, this model is evolving, fortified by the power of AI/ML, to effectively manage complex, dynamic environments and tasks prevalent in next-network generations such as 6G networks. This evolution signifies the shift from static, pre-programmed agents to dynamic, adaptable ones capable of continuous learning - a truer manifestation of intelligence.

This paradigmatic transition is characterized by the infusion of AI/ML into the BDI model. Here, an agent’s beliefs transition from static assertions to predictive models shaped by past experiences. In this enhanced model, the belief function \( B \) can be characterized as a mapping from a set of past experiences \( E \) to a set of future predictions \( P \), formulated as \( B : E \rightarrow P \). This learning-based model facilitates dynamic belief updates as the agent amasses experience, leading to performance optimization over time.

In the context of 6G networks, agents will encounter diverse data types - structured, semi-structured, and unstructured, from various sources. This necessitates the integration of perception, a critical aspect of intelligent systems, into the BDI model. Such integration aids in processing sensory inputs into the agent’s belief function, facilitating a more comprehensive understanding of the network environment and enabling informed, effective actions. Formally, given \( S \) represents the set of sensory inputs and \( B' \) is the updated belief function, the agent’s beliefs can be updated as \( B' : (E \times S) \rightarrow P \).

This paradigm shift to AI/ML in multi-agent systems, particularly for 6G network performance enhancement, marks a fascinating advancement in the field, catalyzing the development and deployment of more autonomous, adaptive, and intelligent agents. However, this transition also uncovers new challenges. For instance, assuring the safe and reliable operation of these learning agents within a multi-agent environment is paramount, further opening up a promising and extensive realm for future exploration.
6.3 SP-LSTM: A Novel Agent in 6G Networks

6.3.1 Introduction to Speed-optimized LSTM (SP-LSTM)

In the complex milieu of artificial intelligence and machine learning algorithms, Long Short-Term Memory (LSTM) networks, a variety of Recurrent Neural Network (RNN), have carved a niche for themselves as effective tools for predictive analytics. These networks, with their capability to decipher long-term dependencies in temporal data, stand as an ideal solution to identify intrinsic temporal patterns in network traffic. This proficiency in leveraging past data to prognosticate future network congestions makes LSTM networks indispensable for proactive network management. By anticipating potential congestion areas, models based on LSTM facilitate proactive data rerouting, thus aiding in congestion avoidance and augmenting network throughput [58].

It is important to comprehend the operations of LSTM networks to truly understand their utility. Given an input sequence denoted by \((x_1, x_2, \ldots, x_t)\), the internal states of an LSTM are updated through the equations illustrated in Equation (6.1):

\[
\begin{align*}
\text{Forget gate:} & \quad f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \\
\text{Input gate:} & \quad i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \\
& \quad \hat{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{6.1} \\
\text{Cell state update:} & \quad C_t = f_t \ast C_{t-1} + i_t \ast \hat{C}_t \\
\text{Output gate:} & \quad o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \\
& \quad h_t = o_t \ast \tanh(C_t)
\end{align*}
\]

In these equations, \(\sigma\) signifies the sigmoid activation function, the \(\tanh\) symbolizes the hyperbolic tangent activation function, the \(\ast\) represents element-wise multiplication, the \([., .]\) stands for concatenation, the \(W\) terms are weight matrices, and the \(b\) terms are bias vectors.

However, the potency of LSTM-based predictive analytics is not devoid of challenges.
The prerequisite of voluminous historical data to train LSTM models necessitates considerations regarding data storage, privacy, and the risk of over-reliance on past network behaviors. Additionally, the effectiveness of LSTM performance could be considerably contingent on the quality of the input data and the selection of appropriate hyperparameters.

While Long Short-Term Memory (LSTM) networks exhibit significant prowess over conventional Recurrent Neural Networks (RNNs) in handling long-term dependencies, they do not come devoid of challenges:

- **Computational Burden**: The intricate gating mechanism of LSTMs heightens their computational demands, decelerating the process of training and inference. This sequential progression of computations can pose a major hurdle in real-time applications.

- **Training Complexities**: The elaborate structure and multitude of gates in LSTMs exacerbate the complexities of training. Although less frequent than basic RNNs, LSTMs can also fall victim to the vanishing or exploding gradient problem.

- **High Parameter Volume**: The massive parameter count associated with LSTMs can foster overfitting, particularly in scenarios with limited datasets.

- **Memory Requirement**: The necessity to store numerous parameters and interim cell states for backpropagation can impose a substantial memory burden, a crucial factor to consider for hardware implementations.

- **Limited Interpretability**: Much like many neural networks, LSTMs often function as ‘black boxes’, obscuring the understanding of their internal decision-making processes, thus raising issues of transparency and trust.

- **Challenges with Extraordinarily Long Sequences**: Albeit designed to process longer sequences, LSTMs can struggle with extremely elongated sequences, leading
to the loss or distortion of information over time [68].

Notwithstanding these limitations, LSTMs have exhibited exceptional capabilities in tasks concerning sequential or time series data. Furthermore, advancements are continually being made to augment their performance and alleviate these issues. Alternatives such as Gated Recurrent Units (GRUs) or the Transformer model (utilized in BERT and GPT) present different trade-offs [154]. To address these challenges more comprehensively, the 'Speed-optimized LSTM' (SP-LSTM), a revolutionary enhancement of the traditional LSTM, is proposed as a major contribution of this thesis.

This work introduces a revolutionary model that considerably strengthens the LSTM framework by incorporating a range of crucial enhancements. Initially, the Speed-optimized LSTM (SP-LSTM) streamlines the conventional four-gate architecture of LSTM by converging the forget and input gates into a unified 'update gate'. This consolidation not only mirrors the efficiency of the Gated Recurrent Unit (GRU) but also maintains the distinct capacity of LSTM to handle extended sequence tasks.

Subsequently, to combat the sequential characteristic of LSTM that impedes temporal parallelization, SP-LSTM cleverly incorporates elements from the Transformer’s self-attention mechanism [154]. This integration empowers our model to assess all temporal steps concurrently, understanding their relevance, thereby optimally allocating computational resources. In response to the issue of overfitting in LSTM due to its voluminous parameter count, SP-LSTM embeds regularization techniques directly into its structure. This involves integrating dropout layers within LSTM cells and implementing effective strategies like Batch Normalization, reducing the propensity for overfitting while preserving model performance [72].

Our innovative SP-LSTM model, equipped with an explicit memory mechanism inspired by 'memory networks' or the 'Neural Turing Machine' concept, proficiently manages exceedingly long sequences [59]. This significant upgrade propels LSTM’s efficiency in tasks involving extended sequential data to new heights. Finally, SP-LSTM instills
the much-needed attribute of explainability into the LSTM sphere. Utilizing attention mechanisms, our model transcends mere prediction and offers insightful explanations by emphasizing parts of the input it concentrates on for its predictions.

In encapsulating the key concepts of our contribution, we present the robust and efficient SP-LSTM in an algorithmic form (Algorithm 11). This pioneering methodology simplifies processing and propels machine-learning algorithms to the upcoming frontier of speed and precision. The algorithm 11 and Table 6.1 effectively synthesize the intricacies of SP-LSTM into an understandable format, thereby serving as a valuable tool for understanding and employing this potent AI/ML paradigm in the realm of 6G network management and beyond.

Algorithm 11 Pioneering SP-LSTM Framework for Anticipating Network Bottlenecks

1: procedure SP-LSTM(X)
2:     Start with $C[0] = 0, h[0] = 0$
3:     for each $t$ within the sequence do
4:         Determine $u[t] = \sigma(W_u[h[t-1],X[t]] + b_u)$
5:         Calculate $\hat{C}[t] = \tanh(W_c[h[t-1],X[t]] + b_c)$
6:         Refresh $C[t] = u[t] \cdot C[t-1] + (1 - u[t]) \cdot \hat{C}[t]$
7:         Compute $a[t,i]$ for all $i$, standardize to get $a[t,:]
8:         Evaluate $\hat{C}[t] = \Sigma_i(a[t,i] \cdot C[i])$
9:         Calculate $o[t] = \sigma(W_o[h[t-1],X[t],\hat{C}[t]] + b_o)$
10:        Refresh $h[t] = a[t] \cdot \tanh(\hat{C}[t])$
11:    end for
12:    output the full sequence $h[0], ..., h[T]$ if return_sequences=True, else the final state $h[T]$ only.
13: end procedure

Where:

- $h[t]$: The concealed state at the $t^{th}$ instant.
- $x[t]$: The received input at the $t^{th}$ moment.
- $C[t]$: The cell’s state at the $t^{th}$ time point.
• $u[t]$: The modified gate at the $t^{th}$ timestamp.

• $o[t]$: The exit gate at time $t$.

• $W_u, W_c, W_o$: The respective weight matrices for the updated gate, substitute memory cell, and output gate.

• $b_u, b_c, b_o$: The respective bias vectors for the updated gate, substitute memory cell, and output gate.

• $\sigma$: The sigmoid operation.

• $\tanh$: The hyperbolic tangent operation.

• $a[t, i]$: The attention metric at the $t^{th}$ time point for cell state $i$.

• $\tilde{C}[t]$: The tentative cell state at the $t^{th}$ instance.

• $\text{return\_sequences}$: A logical operator that defines whether to output the hidden states for all time steps or only the last one.

A meticulous time complexity analysis of the Pioneering SP-LSTM Framework for Anticipating Network Bottlenecks is elucidated in Appendix A. This thorough exploration provides a profound understanding of the system’s performance, subsequently serving as a pivotal resource for refining future versions of the system.
Consolidated Gate Structure: Merges the forget and input gates into a unified 'update gate', enhancing efficiency without compromising LSTM’s proficiency for long sequence tasks.

Incorporation of Self-Attention: Embraces components from the Transformer’s self-attention mechanism, allowing simultaneous assessment of all time steps, leading to optimal utilization of computational resources.

Embedded Regularization: Incorporates dropout layers within LSTM cells and applies Batch Normalization techniques, decreasing the risk of overfitting while preserving model performance.

Explicit Memory Mechanism: Employs a memory mechanism reminiscent of 'memory networks' or the Neural Turing Machine concept, offering effective handling of extremely long sequences.

Inclusion of Explainability: Harnesses attention mechanisms, enabling the model to not only forecast but also provide insightful explanations by emphasizing the input segments it pays attention to during predictions.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Consolidated Gate Structure</td>
<td>Merges the forget and input gates into a unified 'update gate', enhancing efficiency without compromising LSTM’s proficiency for long sequence tasks.</td>
</tr>
<tr>
<td>Incorporation of Self-Attention</td>
<td>Embraces components from the Transformer’s self-attention mechanism, allowing simultaneous assessment of all time steps, leading to optimal utilization of computational resources.</td>
</tr>
<tr>
<td>Embedded Regularization</td>
<td>Incorporates dropout layers within LSTM cells and applies Batch Normalization techniques, decreasing the risk of overfitting while preserving model performance.</td>
</tr>
<tr>
<td>Explicit Memory Mechanism</td>
<td>Employs a memory mechanism reminiscent of 'memory networks' or the Neural Turing Machine concept, offering effective handling of extremely long sequences.</td>
</tr>
<tr>
<td>Inclusion of Explainability</td>
<td>Harnesses attention mechanisms, enabling the model to not only forecast but also provide insightful explanations by emphasizing the input segments it pays attention to during predictions.</td>
</tr>
</tbody>
</table>

Table 6.1. Principal Attributes of the Speed-Optimized LSTM

6.3.2 Role and Performance of SP-LSTM in Multi-Agent Systems

In the realm of autonomous network systems, the significance of multi-agent systems (MAS) cannot be understated. These distributed systems, encompassing multiple inter-
acting agents, provide an excellent framework for modeling and solving complex problems that are beyond the capacity of individual agents. The potency of a MAS, however, hinges upon the sophistication of its constituent agents’ decision-making processes. To that end, the SP-LSTM—Speed-optimized Long Short-Term Memory—emerges as an innovative player, its design tailored to handle the dynamic nature and temporal dependencies inherent in MAS-based operations.

The primary responsibility of the SP-LSTM within MASs revolves around its advanced predictive and learning capabilities. With the capacity to process long sequences of data, the SP-LSTM aids the agents in understanding the temporal patterns within their environment. By doing so, it enables the agents to not only react intelligently to the current state of the environment but also anticipate future states, effectively bolstering the system’s overall performance.

As illustrated in the subsequent section, our experimental evaluation of the SP-LSTM implementation within a MAS for autonomous network management manifests its potential. Compared to traditional LSTMs and other recurrent neural network (RNN) models, the SP-LSTM yielded superior results in terms of accuracy and computational efficiency. By capitalizing on the SP-LSTM’s enhanced temporal comprehension and parallelizable architecture, the MAS demonstrated an improved capacity to manage network congestion and predict potential bottlenecks.

Moreover, the SP-LSTM’s in-built regularization mechanisms and the addition of an explicit memory component offered notable improvements. The propensity for overfitting, a typical predicament in LSTMs due to their high parameter count, was substantially reduced. Simultaneously, the ability to effectively handle extremely long sequences augmented the system’s proficiency in tracking and predicting network states over an extended period.

Equally imperative is the element of explainability introduced by SP-LSTM. By leveraging attention mechanisms, the agents in the MAS can now provide insightful explana-
tions about their predictions. This layer of interpretability is instrumental in debugging, improving, and, importantly, trusting the system in critical real-world applications.

The SP-LSTM emerges as a formidable component in the design of intelligent and robust multi-agent systems for autonomous network management. Its advanced architecture not only amplifies the system’s performance but also proffers insights into the decision-making processes, reinforcing the reliability of MAS in dynamic and complex network environments.

6.4 Reinforcement Learning: An Interactive Agent for Dynamic Routing

6.4.1 The Deployment of Reinforcement Learning within the Context of Autonomous Agents

With the proliferation of autonomous systems in network management, automation of hitherto manual tasks is becoming a new standard. A key driver of this transformative change is the advent of autonomous agents, empowered to learn autonomously and make informed decisions. The cornerstone of such autodidactic abilities lies in reinforcement learning (RL), a robust method that enables these agents to learn from their surroundings and improve their performance iteratively [144].

Reinforcement learning, an integral pillar of contemporary machine learning, employs the mathematical framework of Markov decision processes (MDPs) to enable an agent to interact with its environment [66]. The essence of RL lies in learning a policy \( \pi : S \rightarrow A \), where \( S \) is the set of states and \( A \) is the set of actions, that maximizes the expected cumulative reward. This learning paradigm is particularly fitting for multi-agent systems (MAS), which often operate in volatile and unpredictable environments.

Within the context of autonomous network management, RL-empowered agents within an MAS navigate through a vast state space, encompassing the different network states. They evaluate the impact of their actions, iteratively refining their policies \( \pi \) to minimize network congestion and balance network load. The learning process, inherently a
stochastic approximation [27], enables the agents to amass experience and hone sophisticated strategies that incorporate not only current network states but also foresee future states based on historical data.

Notably, RL agents are intrinsically goal-oriented, acting as optimizers of a cumulative reward function. They adapt their strategies according to a balance of exploration and exploitation, giving precedence to long-term rewards over immediate gains. This philosophy aligns seamlessly with network management scenarios, where today’s actions could dictate the network’s future performance.

A pivotal strength of RL is its harmonious integration with function approximators such as neural networks. By unifying RL with deep learning—a synergy known as deep reinforcement learning—the handling of high-dimensional data becomes feasible, leading to improved management of large-scale network environments.

Reinforcement learning equips agents in multi-agent systems with a robust mathematical apparatus to learn effectively from and interact with their environments. By endorsing continuous learning, goal-centric decision-making, and dynamic strategy development, RL lays the groundwork for autonomous network systems capable of managing complex, dynamic network scenarios.

6.4.2 Interaction of Reinforcement Learning with Other Agents in the System

As autonomous network systems grapple with the complexities of fluctuating network conditions, it becomes apparent that historical data, although valuable, may not suffice to predict future network states accurately. An amalgam of prospective and responsive measures is therefore crucial to proficiently manage the dynamic network environment.

This research advocates for a symbiotic union of SP-LSTM-enabled predictive analytics and dynamic routing techniques to address this challenge. Predictive analytics, driven by SP-LSTM’s temporal learning capabilities, offer preliminary insights that guide the
initial routing decisions. This role of predictive analytics stands as a proactive measure, establishing an initial state of network optimization.

Subsequently, the application of dynamic routing techniques ensures that real-time adaptations occur in response to live network states, thereby introducing a reactive element to the management process. Dynamic routing, a vital constituent of advanced network management techniques, leverages AI/ML algorithms to facilitate real-time adjustments in the routing paths based on live network conditions.

This integrative approach harmonizes the predictive and responsive measures, augmenting the robustness of network performance against both anticipated and unanticipated changes. The synergy between SP-LSTM-based predictive analytics and dynamic routing expands the realms of possibilities within network management, offering an advanced solution that intelligently adapts to evolving network scenarios.

In the subsequent section, we further explore AI/ML-powered dynamic routing and its transformative potential when allied with predictive analytics in autonomous network management.

6.5 Pioneering Network Management through Predictive Intelligence and Dynamic Adaptation in Multi-Agent Systems

In the diverse spectrum of network management strategies, dynamic routing emerges as a linchpin for network efficiency, signifying an epochal shift from static, inflexible routing strategies to a dynamic, real-time adaptable model.

The advent of AI/ML technologies has been a game-changer, amplifying the effectiveness of routing processes. Notably, reinforcement learning (RL) - a promising branch of machine learning, illustrates considerable potential for integration into dynamic routing applications. RL, with its unique amalgamation of interaction and learning processes, facilitates the development of optimal strategies through an iterative feedback loop.

RL bestows agents within the realm of dynamic routing the ability for incessant
learning, enabling them to continually optimize routing policies. In this feedback-rich environment, the RL agent, dictated by the current network state, makes decisions, each of which is followed by feedback reflecting the repercussions of these decisions on overall network performance. This cyclical interaction propels the agent to continually refine its routing strategy, thereby improving its decision-making capabilities over iterations.

The crux of an RL-integrated routing system lies in the optimization of a reward function that represents the ideal network conditions. The agent’s performance is gauged based on key metrics such as latency reduction, congestion minimization, and equitable network load distribution. It is thus incentivized for performance-enhancing actions and penalized for those contributing to performance deterioration.

This strategic objective is encapsulated in Equation (6.2):

\[
\max_{\pi} \mathbb{E}_{(s,a) \sim \pi}[R(s,a)]
\] (6.2)

Here, \( \pi \) refers to the routing policy. This policy is a key determinant of the agent’s behavior in the network, guiding how routing decisions are formulated based on the network’s current state. Mathematically, this policy is a function mapping a network state \( s \) to the probabilities of choosing each possible action \( a \). Consequently, \( \pi(a|s) \) signifies the probability that the agent will opt for action \( a \) in state \( s \). The function \( R(s,a) \) encapsulates the reward mechanism. We calculate the expected value, denoted as \( \mathbb{E} \), based on the sequence of state-action pairs yielded by faithfully implementing the policy \( \pi \).

The chosen action could encompass various operations ranging from selecting a specific data transmission path to switching nodes. The optimal policy, denoted by \( \pi^* \), is the one that maximizes the expected cumulative reward over all states, leading to the most efficient routing decisions.

Notably, the policy \( \pi \) can be either deterministic, implying a fixed action for each state, or stochastic, indicating an action selected based on a probability distribution.
The preference between deterministic and stochastic policies hinges on the specificities of the network and the routing process requirements.

Unique to RL, the policy $\pi$ isn’t static but undergoes refinement as the agent learns from the environment. This continual evolution is made possible by updating the policy grounded in the rewards and penalties accrued due to the actions taken, thus honing the decision-making process in response to network states and their consequent implications.

The infusion of RL into dynamic routing harnesses the formidable power of AI/ML technologies to conceive a robust, adaptable, and efficient network management system. When synergized with predictive analytics driven by SP-LSTM, this potent combination can proficiently navigate both predictable and unpredictable network conditions. It marks a new era in network management, wherein predictive intelligence is employed not merely for forecasting but to proactively shape future network states, thereby revolutionizing traditional network paradigms.

6.6 Innovating Network Management: A Fusion of Predictive Intelligence and Dynamic Adaptation

The architecture proposed herein represents a revolutionary approach to network management, meticulously tailored to surmount the intricate challenges of upcoming generation networks. By leveraging the potent predictive capabilities of SP-LSTM-based analytics and the dynamic flexibility of RL-based routing, we engineer a bifurcated AI/ML structure that blends the strengths of both domains, thereby constructing a solution that not only ensures efficiency but also robustness amidst variable network conditions.

The initial layer of our system employs SP-LSTM networks, acclaimed for their proficiency in analyzing time-series data. Utilizing historical network data, the SP-LSTM model performs predictive analyses of forthcoming network conditions, thereby identifying potential congestion hotspots and facilitating preemptive decisions regarding data rerouting and load balancing. SP-LSTM utilizes a sequence of past network states
(x_1, x_2, \ldots, x_t) to predict the upcoming network state x_{t+1}, as represented in Equation (6.3):

\[ x_{t+1} = SP-LSTM(x_1, x_2, \ldots, x_t) \] (6.3)

Therefore, SP-LSTM-assisted predictive analytics establishes a primary routing strategy, based on forecasted network conditions.

Further augmenting this foundation, the secondary layer employs reinforcement learning to dynamically fine-tune routing decisions. Grounded in the philosophy of learning through experience, RL algorithms, such as Q-learning or SARSA, utilize real-time network feedback to continuously calibrate and optimize the routing policy \( \pi^* \) in a bid to maximize the overall expected reward over time, as described in Equation (6.4):

\[ \pi^* = \arg \max_\pi \mathbb{E} \left[ \sum_{t=0}^{T} R(s_t, a_t) \right] \] (6.4)

In this equation, \((s_t, a_t)\) represents the state-action pairs (i.e., network state and routing action) at time step \(t\), while \(R(s_t, a_t)\) denotes the reward function, which evaluates the suitability of a particular state-action pair.

To elucidate further, the RL framework is built upon the Markov Decision Process (MDP) paradigm. Technical details of the RL implementation can be found in Table 6.2 and the accompanying GitHub repository. The essence of the MDP consists of states, actions, transitions, and rewards. Within our system:

- **State Space (S):** Every state \(s \in S\) is a vector representation that includes features such as Link ID, Time, and various throughput metrics. More specifically, given a network with \(N\) links, and assuming we’re considering \(T\) time steps for throughput metrics, the state at any time \(t\) can be represented as \(s_t = [\text{link}_{1\ldots N}, \text{time}_{1\ldots T}, \text{throughput}_{1\ldots T}]\).

- **Action Space (A):** The action \(a \in A\) corresponds to routing decisions made
Based on the current network topology. Given a device $d$ in the network, actions can involve routing to any of the directly connected links. Thus, for a device with $L$ connected links, the action space will have $L$ possible routing decisions.

- **Transition Probabilities (P):** These define the likelihood $P(s_{t+1}|s_t, a_t)$ of transitioning from one state $s_t$ to another $s_{t+1}$ given an action $a_t$. In our network scenario, these probabilities can be estimated using historical data combined with real-time feedback. For instance, given the current state of the network and a chosen routing action, we can estimate the probability that a certain link will be congested in the next time step.

- **Reward Function (R):** As previously mentioned, our reward function links the predicted moving average throughput of selected links to potential rewards for respective routing actions. Mathematically, for a given state-action pair, $R(s_t, a_t)$ quantifies the benefit of routing through a particular link given the current network conditions.

<table>
<thead>
<tr>
<th>Table 6.2. Q-Learning Architecture for Routing Decisions</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Component</strong></td>
</tr>
<tr>
<td>States</td>
</tr>
<tr>
<td>Actions</td>
</tr>
<tr>
<td>Reward</td>
</tr>
<tr>
<td>$\alpha$</td>
</tr>
<tr>
<td>$\gamma$</td>
</tr>
<tr>
<td>$\epsilon$</td>
</tr>
<tr>
<td>Convergence</td>
</tr>
<tr>
<td>Episodes</td>
</tr>
<tr>
<td>Q-Table</td>
</tr>
</tbody>
</table>

Through its continuous learning and adaptability, the RL-based dynamic routing layer
counteracts any inherent limitations of LSTM-based predictions, ensuring effective routing decisions even when faced with unpredictable network changes. The introduction of RL into future-generation networks could bring about significant transformation, largely due to its inherent emphasis on maximizing long-term benefits, diverging from traditional models focused on short-term gains. This fosters the incorporation of forward-looking mechanisms within future network frameworks. Furthermore, RL considerably trims signaling overhead costs by facilitating autonomous learning from the network environment, leading to a streamlined approach that boosts network performance and reduces operational expenditure, thereby advocating for its integration into leading-edge Next-generation networks.

![AI/ML-Powered Predictive and Dynamic Routing System](image)

**Figure 6.3: AI/ML-Powered Predictive and Dynamic Routing System.**

Our groundbreaking strategy introduces a sophisticated bifurcated AI/ML system, with the potential to redefine the management of emergent 6G networks. Fig. 6.3 offers a comprehensive visual portrayal of the proposed system, highlighting the workflow and synergistic interactions among its various constituents:

- **Data Collection**: Signifies the data sources, comprising historical network data for SP-LSTM analytics and real-time state data for RL routing.

- **SP-LSTM-driven Predictive Analytics**: Leverages historical data to anticipate future network states.
• **Projected Network States**: Represents the predicted network state derived from historical data.

• **RL-powered Dynamic Routing**: Receives input from both Projected Network States and real-time state data, fine-tuning the routing policy based on real-time feedback and projected conditions. It underscores continuous learning and adaptation for dynamic optimization.

• **Routing Policy**: Depicts the output of RL Dynamic Routing, namely the optimal routing policy at a particular time.

• **Network Administration**: The final phase, which enforces the derived routing policy in the network.

Balancing proactivity in forecasting with reactivity in adaptation, this system can improve various network performance indicators, including latency and congestion, thereby providing a promising solution to meet the diverse and stringent demands of 6G services.

### 6.7 The Potential of Multi-Agent Systems to Outperform Traditional Network Management Techniques

In a bid to validate the theoretical underpinnings of our proposed architecture, an illustrative use case is presented that underscores the transformative capabilities of a proactive AI/ML system for resource optimization and performance enhancement in the context of 6G networks. By harnessing the power of cutting-edge technologies such as predictive analytics and dynamic routing, which are fueled by SP-LSTM networks and RL, we facilitate complex network pattern modeling, precise congestion prediction, and proactive circumvention of disruptions.

The use case demonstrates the effectiveness of utilizing real-time data in conjunction with intelligent routing decisions, thereby assuring high network performance. It
aptly handles the massive data payloads characteristic of 6G networks and enables near-instantaneous routing decisions. The results obtained from the use case substantiate the practical applicability of the proposed framework, and the mathematical formulation for the same can be represented as follows:

Assuming the network state at time $t$, $x_t$, and the action, $a_t$, the predictive SP-LSTM model for the next network state can be represented as:

$$x_{t+1} = SP-LSTM(x_t, a_t)$$ (6.5)

While the RL model for the optimal action can be expressed as:

$$a_{t+1} = \arg \max_a \mathbb{E} \left[ \sum_{t=0}^{T} R(x_t, a_t) \right]$$ (6.6)

These equations form the core of the decision-making process, enabling the system to respond proactively and dynamically to the ever-changing network conditions.

The entire code base supporting this use case, providing a tangible demonstration of our resource optimization strategy, can be retrieved from the following online repository: GitHub. The integration of such a system within the framework of 6G networks can mark a pivotal step towards more efficient, resilient, and responsive network management.

### 6.7.1 Network Topology

The conceived network configuration in our research has been intricately designed utilizing NetSim v13.3 [146], catering explicitly to the nuanced requirements of 6G networks. The graphical representation of the configuration, as depicted in Figure 6.4, is characterized by nodes, interconnections, and quintessential elements of 5G and 6G networks. Key components such as the AMF for managing access and mobility, the SMF for handling session management and traffic optimization, UPF for processing user plane data traffic, and gNB acting as the base station for signal transmission/reception and radio resource
management, have been included in the topology. Furthermore, an array of mobility models has been applied to user equipment (UE) to simulate realistic mobility scenarios.

![NetSim designed topology mimicking a 6G network infrastructure](image)

The network can be modeled as a graph $G = (V, E)$ where $V$ denotes the set of nodes in the network, and $E$ represents the set of links connecting these nodes. The specific components of the network can be denoted as $V = v_{AMF}, v_{SMF}, v_{UPF}, v_{gNB}, v_{UE}$ and $E = (v_i, v_j) | v_i, v_j \in V$. This representation enables an easier understanding of the network’s structure and its various connections.

Essential characteristics and technologies emblematic of 5G and 6G networks, such as Massive MIMO, mmWave communications, network slicing, edge computing, SDNs, and VNFs, have been integrated into the topology. This amalgamation facilitates a robust simulation of these networks and allows for a comprehensive evaluation of network management and optimization strategies that are targeted to tackle the distinct needs and challenges presented by 6G networks.

The experimental setup is further enriched with realistic traffic patterns and an as-
sortment of traffic types to enhance authenticity. These patterns mirror the attributes of nascent 6G applications and services, which range from ultra-high-definition video streaming to IoT device communication and mission-critical communications. The heterogeneous traffic types provide an opportunity to assess the network’s ability to effectively manage the stringent requirements of various use cases and applications.

The network topology has been meticulously designed to mimic real-world 6G networks in terms of scale, heterogeneity, and connectivity, thereby ensuring the relevance and applicability of our research.

6.7.2 Dataset

Harnessing the detailed network topology defined in the preceding section, an extensive dataset has been compiled, enriched with fundamental features pivotal for our investigative study. The constructed dataset incorporates crucial attributes such as Connection_Reference, Timestamp, Immediate_Throughput, Sliding_Window_Throughput, and Cumulative_Average_Throughput.

The attribute Connection_Reference acts as an integral identifier, signifying the network linkage among distinct nodes within the established topology, thereby laying a solid base for future analyses. The Timestamp feature, on the other hand, reflects the temporal dimension of the data, paving the way for the exploration of temporal correlations.

The Sliding_Window_Throughput offers an understanding of the moving average throughput over time, which is beneficial for tracking long-term trends and identifying patterns in the data. The Immediate_Throughput portrays real-time fluctuations in network performance, allowing for precise measurement of instant network efficiency.

In contrast, the Cumulative_Average_Throughput encapsulates the average throughput over time, providing an overall picture of the network’s operational efficiency. By embedding these key features into our dataset, our research is fortified with a compre-
hensive resource that encourages exhaustive exploration and in-depth analysis of the intricacies of network dynamics and performance.

6.7.3 Predictive Analytics and Congestion Analysis

Commencing with the importation of KeyDataset.csv, the process deploys integral attributes namely, Immediate_Throughput, Connection_Reference, Timestamp, Sliding_Window_Throughput, and Cumulative_Average_Throughput. Pre-processing commences with data standardization employing MinMaxScaler and subsequently dividing the set into training and testing subsets. The SP-LSTM model is invoked to seize temporal correlations and predict Sliding_Window_Throughput.

Evaluation of the model’s efficacy employs the mean squared error (MSE) and determines the prediction accuracy. The predicted outputs are archived as CSV files for supplementary analyses. Further, an assessment of congestion levels in network links is undertaken based on Sliding_Window_Throughput prior and subsequent to prediction, illustrating the effectiveness of our method. Our evaluative techniques offer quantitative evidence into the efficacy of our solution for latency minimization and the enhancement of network performance in diverse 6G landscapes.

6.7.4 AI/ML-Based Strategies for Optimal Network Management

Embarking on a mission for resilient, data-driven network administration, our investigation uncovers a pioneering AI/ML-oriented system (refer to Algorithm 12). This is juxtaposed against a conventional network management scheme (detailed in Algorithm 13), enabling a direct efficacy comparison under equivalent network conditions.

Algorithm 12 outlines an AI/ML methodology employing Q-learning, a Reinforcement Learning (RL) variant, to streamline network administration. It assimilates states, actions, and rewards inferred from the network topology and Forecasted Moving Average throughput (elucidated in section 6.7.2). This algorithm equips the AI to ascertain
the optimal data transmission route by exploring or exploiting the network. The chosen pathway aims to minimize the variance between predicted and current moving average throughput, thus augmenting network performance. The Q-table, dynamically updated with each episode, retains the Q-values for all states, steering the model toward the optimal route. Additional facets encompass network topology visualization, optimal path emphasis, and monitoring rewards and path length evolution over episodes.
Algorithm 12 Q-Learning Approach for Optimal Network Path Selection

1: procedure QLEARNINGPATHSELECTION(DataFrame, NetworkTopology)
2:     Load DataFrame, NetworkTopology
3:     Initialize QTable, learning rate ($\alpha$), discount factor ($\gamma$), exploration rate ($\epsilon$), number of episodes (numEpisodes), Graph (G), and state variables
4:     for episode in range (1, numEpisodes) do
5:         Initialize state, rewardSum, and pathLength
6:         while not terminal state do
7:             action $\leftarrow$ EpsilonGreedyPolicy(QTable, state, $\epsilon$)
8:             nextState $\leftarrow$ TakeAction(action)
9:             reward $\leftarrow$ GetReward(nextState)
10:            Update QTable[state, action] using Q-learning update rule:
11:               QTable[state, action] $\leftarrow$ QTable[state, action] + $\alpha$ * (reward + $\gamma$ * max(QTable[nextState]) - QTable[state, action])
12:         Update current state: state $\leftarrow$ nextState
13:         if state == 'gNB,7' then
14:             Break loop
15:         end if
16:     end while
17:     Update rewardSum, pathLength for current episode
18:     if episode % 1000 == 0 then
19:         Print progress report
20:     end if
21: for
22:     Print summary of episode values
23:     Compute optimalPath, totalWeight from QTable
24:     Visualize NetworkTopology with optimalPath
25:     Plot rewardSum and pathLengths across episodes
26:     Print mean(rewardSum), mean(pathLengths) across all episodes
27: end procedure

Conversely, Algorithm 13 portrays OSPF, a traditional network management technique utilizing Dijkstra’s shortest path algorithm. It generates a graph based on network
topology data, where edge weights symbolize the reciprocal of anticipated throughput. Dijkstra’s algorithm identifies the shortest path, serving as the optimal data transmission route. This approach provides comprehensive data examination tools, inclusive of network topology visualizations with the optimal path and a histogram of forecasted throughput, as well as the distribution of predicted throughput across links.

The comparison of these methodologies delivers pivotal insights. The findings reveal that while both methodologies can detect optimal paths, the AI/ML-based approach (Algorithm 12) displays superior adaptability and performance in dynamic network settings. The algorithm’s ability to learn from experiences and adapt strategies over time offers a significant edge, underlining the substantial potential of AI/ML in revolutionizing network administration practices.

**Algorithm 13** Network Graph Shortest Path Calculation

1: `procedure COMPUTE_SHORTEST_PATH(Data, PredictedOutput, NetworkTopology)`
2:  `Import libraries: pandas, networkx, matplotlib.pyplot`
3:  `Load Data, PredictedOutput, and NetworkTopology from CSV files`
4:  `Merge Data and PredictedOutput based on Link_ID and Time to form MergedData`
5:  `Create an empty graph, G`
6:  `Add nodes to G from NetworkTopology`
7:  `for each row in MergedData do`
8:  `Extract link_id, predicted_throughput from the row`
9:  `Retrieve associated devices from NetworkTopology based on link_id`
10:  `if devices exist in NetworkTopology then`
11:  `Extract device_name_1, device_name_2 from devices`
12:  `Add an edge to G between device_name_1 and device_name_2 with weight as the reciprocal of predicted_throughput`
13:  `end if`
14:  `end for`
15:  `Define start_node and end_node`
16:  `if start_node and end_node exist in G then`
17:  `Compute shortest path in G using Dijkstra’s algorithm and store it in path`
18:  `Output path`
19:  `end if`
20: `end procedure`
6.7.5 Scalable Prediction of Network Congestion: A Mathematical Perspective

In an endeavor to develop a scalable congestion prediction model for large networks, we architected the advanced SP-LSTM network with a strategic focus on interpreting historical throughput data. Our systematic procedure involves stages from data collection to its preprocessing, model formulation, rigorous assessment, and efficient delivery of the forecasts. A comprehensive summary of the process outcomes is depicted in Table 6.3.

By assessing the model’s training performance, an accuracy of approximately 0.9822 was realized. This substantial figure illustrates the model’s proficiency in comprehending the fluctuations in network throughput, accounting for more than 98% of the data variation (Eq. 6.7). The subsequent Mean Squared Error (MSE) of approximately 1.04132e-07 on the validation set verifies the model’s robust forecasting ability, with the predictions mirroring the real-time values closely (Eq. 6.8).

\[
\text{Accuracy} = \frac{\Sigma(PredictedValue - ActualValue)^2}{n} \quad (6.7)
\]

\[
MSE = \frac{1}{n} \Sigma(PredictedValue - ActualValue)^2 \quad (6.8)
\]

Furthermore, the model’s prediction accuracy approximates to 0.999988 (almost 99.9988%), accentuating its exceptional ability to generalize, thereby performing remarkably well on novel data. This strong performance attests to the model’s capability as a robust and dependable tool for anticipating network congestion.
Table 6.3. Performance Metrics of the SP-LSTM Model

<table>
<thead>
<tr>
<th>Performance Metric</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training Accuracy</td>
<td>0.98220</td>
</tr>
<tr>
<td>Mean Squared Error</td>
<td>1.04132e-07</td>
</tr>
<tr>
<td>Prediction Accuracy</td>
<td>0.99998</td>
</tr>
</tbody>
</table>

6.7.6 SP-LSTM: A Pioneering Stride in 6G Network Analytics

Fig. 6.5 and Table 6.4 lay out a captivating portrayal of a comparative exploration between the conventional LSTM model and the SP-LSTM model, elucidating their ramifications on prospective network management strategies. The SP-LSTM model sets itself apart with its rapid model training and inference phases, indicating a proficient AI/ML-based solution apt for the stringent demands of 6G network systems.

The LSTM model exhibits excellent forecasting accuracy, yet the transformative attribute is the expedited performance of the SP-LSTM model. Its nimbleness during both training and forecasting stages amplifies its aptness for real-time network management scenarios where response time is crucial.
The amplified speed of the SP-LSTM model augments system agility, a vital feature for dynamic routing efficiency. Rapid and optimal routing decisions play an indispensable role in Ultra-Reliable and Low-Latency Communication (URLLC), a key usage scenario in 6G networks. These enhancements indicate a significant leap towards autonomous network management using multi-agent systems, supporting the optimization of real-time communications and network performance.

6.7.7 Analytical Resilience: Decrypting Optimal Network Path and Predicting Congestion

Analyzing the transmutation in ordering of links based on Moving_Average_throughput before and after prediction (Table 6.5) provides crucial inputs for improving strategies in network management. Network management paradigms can be pivoted to a more proactive stance by anticipating future congestion scenarios, which is made possible by these changes in ordering. This was demonstrated in a case study using a Q-learning
Table 6.4. Comparison between Traditional LSTM and SP-LSTM

<table>
<thead>
<tr>
<th>Model</th>
<th>Trial</th>
<th>Training Duration (s)</th>
<th>Prediction Duration (s)</th>
<th>Training Accuracy</th>
<th>MSE</th>
<th>Prediction Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Traditional LSTM</td>
<td>1</td>
<td>316.15</td>
<td>42.24</td>
<td>0.982</td>
<td>2.29e-07</td>
<td>0.99997</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>325.11</td>
<td>27.28</td>
<td>0.982</td>
<td>1.06e-07</td>
<td>0.9999</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>344.55</td>
<td>30.13</td>
<td>0.982</td>
<td>1.15e-07</td>
<td>0.9999</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>340.92</td>
<td>31.34</td>
<td>0.982</td>
<td>4.95e-07</td>
<td>0.99994</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>339.51</td>
<td>31.84</td>
<td>0.982</td>
<td>1.25e-07</td>
<td>0.9999</td>
</tr>
<tr>
<td>SP-LSTM</td>
<td>1</td>
<td>184.34</td>
<td>22.30</td>
<td>0.983</td>
<td>4.16e-05</td>
<td>0.99530</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>164.44</td>
<td>27.60</td>
<td>0.976</td>
<td>1.20e-05</td>
<td>0.99864</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>146.18</td>
<td>21.63</td>
<td>0.981</td>
<td>4.09e-05</td>
<td>0.99537</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>178.62</td>
<td>21.58</td>
<td>0.983</td>
<td>2.33e-05</td>
<td>0.99736</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>145.82</td>
<td>21.51</td>
<td>0.981</td>
<td>1.15e-05</td>
<td>0.99870</td>
</tr>
</tbody>
</table>

algorithm with AI/ML reinforcement, where an optimal network path was identified in a complex telecom scenario of data transmission from gNB_18 to gNB_8.

In this setting, the efficiency of selected paths was quantified by the parameters **Sum_of_Rewards** and **Path_Length**. Here, **Sum_of_Rewards** can be seen as a running tally of gains an agent achieves while traversing the network, which indicates the efficacy of path selection. In contrast, **Path_Length**, representing the count of steps from the start node to the target node, should ideally be minimized to optimize speed and resource consumption. Under latency-sensitive conditions, **Path_Length** may supersede **Sum_of_Rewards** in importance.

Insights derived from results analysis (Figure 6.7 and Table 6.6) demonstrate a preponderance of episodes with a null sum of rewards and path length, indicative of a lack of beneficial path discovery. Yet, there exist exceptions, including episodes 14, 28, 42, 56, 60, 67, 73, 89, and 95, where the agent successfully discovered advantageous paths.

The persistence of the Q-learning algorithm in discovering optimal paths (refer to Figure 6.7), despite numerous episodes yielding null rewards, attests to its resilience. Analysis of episodes with non-zero rewards and their associated path lengths provide key insights into the optimization process, aiding future research aimed at bolstering the algorithm’s efficiency. This presents a novel implementation of reinforcement learning for optimal network path identification, implying significant implications for the enhance-
Table 6.5. Network links ranked from most to least congested: Before and After Prediction

<table>
<thead>
<tr>
<th>Rank</th>
<th>Link ID (Before Prediction)</th>
<th>Link ID (After Prediction)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>8</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>13</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>15</td>
</tr>
<tr>
<td>4</td>
<td>13</td>
<td>10</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>14</td>
</tr>
<tr>
<td>6</td>
<td>4</td>
<td>9</td>
</tr>
<tr>
<td>7</td>
<td>15</td>
<td>-</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>-</td>
</tr>
<tr>
<td>9</td>
<td>10</td>
<td>-</td>
</tr>
<tr>
<td>10</td>
<td>7</td>
<td>-</td>
</tr>
<tr>
<td>11</td>
<td>14</td>
<td>-</td>
</tr>
<tr>
<td>12</td>
<td>9</td>
<td>-</td>
</tr>
<tr>
<td>13</td>
<td>6</td>
<td>-</td>
</tr>
</tbody>
</table>

The results summarized in Table 6.7 underline the potential of the Q-learning algorithm in optimizing resource utilization, showcasing an optimal path total weight of approximately 0.202. The algorithm’s ability to maximize rewards on chosen paths is demonstrated by the mean reward of -3.84470e-05, while the average path length of 0.1 shows its aptitude for identifying shorter, more efficient paths with fewer link traversals.

These findings were juxtaposed with a traditional network management strategy using OSPF, which employs Dijkstra’s algorithm. The key distinction here is the dynamic adjustment of weights in the AI/ML-based Q-learning approach to predicted congestion levels, facilitating more adaptive network management. In contrast, OSPF uses static link costs to identify the shortest path.

Utilizing Dijkstra’s shortest path algorithm, the most resource-efficient route from a
defined start to an end node was established, bypassing potential congestion spots (Tables 6.8 and 6.9). The optimal path \([\text{gNB} _{18}, \text{L3\_Switch} _{4}, \text{gNB} _{8}]\), with corresponding weights of 11.23 and 4.44, was comparable to that obtained via the AI/ML Q-Learning approach, validating the robustness of the proposed AI/ML system.

The ability to identify an optimal path with suitable total weight, corresponding to the mean reward and path length metrics, signifies the effectiveness and practicality of the proposed approach in real-world telecommunications scenarios. These insights facilitate further advancements in AI-driven network management and optimization, paving the way for the practical deployment of AI/ML techniques in the upcoming 6G era.

### 6.7.8 Prospects of Multi-Agent Systems in Nurturing Future Network Management

The inevitable evolution of telecommunication networks toward greater complexity and dynamism, as envisaged in the transition to 6G, necessitates the adoption of intelligent, scalable, and adaptable network management strategies. This research has shed light on the potential of multi-agent systems in redefining the approach toward network congestion prediction and resource optimization. In particular, the successful integration of
SP-LSTM and Q-learning algorithms within a multi-agent system framework has demonstrated promising prospects for enhancing the efficiency of future network systems.

As we have seen, our SP-LSTM model not only ensures accurate throughput predictions, but its agile computation aligns well with the demands for real-time responsiveness in next-generation networks. This is pivotal, especially in ultra-reliable low-latency communication (URLLC) scenarios, where expedient data transmission and decision-making are imperative. Meanwhile, the adaptability of the Q-learning algorithm has emerged as a potent tool for dynamic routing, optimizing path selection based on predicted congestion scenarios.

The use of multi-agent systems in this context has unveiled noteworthy benefits. The
ability to distribute computation among multiple agents paves the way for scalable and resilient network management strategies. This will prove vital as networks continue to expand, both in size and complexity, with the proliferation of IoT devices and heterogeneous communication technologies in the 6G era.

The use of reinforcement learning within multi-agent systems has also shown potential for further exploration. One promising direction lies in extending our current model to multi-agent reinforcement learning (MARL), where multiple agents learn concurrently and interactively, adapting to each other’s policies [67]. This approach could enhance the system’s adaptability, improving path optimization and resource allocation amidst dynamic network conditions.

Moreover, the integration of edge computing within the multi-agent system framework could unlock further enhancements. This decentralizes computation, moving it closer to the network edge, thereby reducing latency and improving resource efficiency [138]. The introduction of AI-driven edge nodes into our multi-agent system could open up new possibilities for decentralized, intelligent network management in future telecommunications networks.
Table 6.7. Q-Learning in Action: Streamlining Resource Utilization through Optimal Path Selection

<table>
<thead>
<tr>
<th>Metric</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optimal Path</td>
<td>{gNB_18, L3_Switch_4, gNB_8}</td>
</tr>
<tr>
<td>Weight (gNB_18 to L3_Switch_4)</td>
<td>0.07692</td>
</tr>
<tr>
<td>Weight (L3_Switch_4 to gNB_8)</td>
<td>0.125</td>
</tr>
<tr>
<td>Total Weight</td>
<td>0.20192</td>
</tr>
<tr>
<td>Average reward</td>
<td>−3.844701</td>
</tr>
<tr>
<td>Average path length</td>
<td>0.1</td>
</tr>
</tbody>
</table>

Table 6.8. Optimal Path Analysis

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optimal Path</td>
<td>{gNB_18, L3_Switch_4, gNB_8}</td>
</tr>
<tr>
<td>Weight (gNB_18 to L3_Switch_4)</td>
<td>11.22887</td>
</tr>
<tr>
<td>Weight (L3_Switch_4 to gNB_8)</td>
<td>4.44496</td>
</tr>
<tr>
<td>Total Weight</td>
<td>0.31402</td>
</tr>
</tbody>
</table>

This research underscores the pivotal role multi-agent systems will play in steering the future of network management. Their scalability, adaptability, and intelligence have set the stage for future studies, promising to catalyze significant strides in autonomous network systems for the 6G era and beyond.

6.8 Chapter Summary

Table 6.9. Alternative Paths Analysis

<table>
<thead>
<tr>
<th>Path</th>
<th>Node 1</th>
<th>Node 2</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>{gNB_18, L3_Switch_4, gNB_8}</td>
<td>gNB_18</td>
<td>L3_Switch_4</td>
<td>11.22888</td>
</tr>
<tr>
<td></td>
<td>L3_Switch_4</td>
<td>gNB_8</td>
<td>4.44496</td>
</tr>
<tr>
<td>{gNB_18, L3_Switch_5, gNB_8}</td>
<td>gNB_18</td>
<td>L3_Switch_5</td>
<td>383.97282</td>
</tr>
<tr>
<td></td>
<td>L3_Switch_5</td>
<td>gNB_8</td>
<td>491.75040</td>
</tr>
<tr>
<td>{gNB_18, L3_Switch_6, gNB_8}</td>
<td>gNB_18</td>
<td>L3_Switch_6</td>
<td>139.78600</td>
</tr>
<tr>
<td></td>
<td>L3_Switch_6</td>
<td>gNB_8</td>
<td>178.84749</td>
</tr>
</tbody>
</table>

The chapter delineates the limitations of traditional network management strategies and how AI/ML-agents can address these challenges. It delves into an analytical exploration of SP-LSTM, asserting its superiority over conventional LSTM models. Notably, the integration of AI/ML into the Belief-Desire-Intention (BDI) model, the cognitive framework for intelligent agents, is discussed, emphasizing the transformation it induces in creating dynamic, autonomous agents.

The chapter further illuminates the role of multi-agent systems (MAS) in enhancing the management of 6G networks, highlighting the fusion of MAS with reinforcement learning as a catalyst for efficient and context-aware routing decisions. It articulates the potential of the novel SP-LSTM architecture in mitigating limitations associated with traditional LSTM models, underscoring its critical role in managing network congestion.

The chapter then ventures into an exploration of reinforcement learning’s role within autonomous network management systems. It propounds the effectiveness of integrating SP-LSTM-based predictive analytics with RL-based dynamic routing, leading to improved network performance and adaptability. The integration of RL into dynamic routing and SP-LSTM’s predictive analytics signifies a leap forward in network management, setting the stage for robust, adaptable network systems.

The chapter concludes with an empirical validation of the proposed AI/ML archi-
tecture for network management, underlining the successful integration of SP-LSTM and Q-learning algorithms within a multi-agent system framework. It acknowledges the transformative potential of the proposed model and proposes directions for future work, including multi-agent reinforcement learning (MARL) and edge computing integration into the multi-agent system framework. This chapter marks a significant stride towards the realization of highly advanced, self-adaptive, and intelligent network systems in the 6G era and beyond.
Chapter 7

Conclusion and Future Work

7.1 Introduction and Epilogue of the Research Journey

As we traverse the final juncture of this research odyssey, this ultimate chapter emerges as a crucial piece in the intricate puzzle of this Ph.D. dissertation. It serves as a high-powered lens, scrutinizing and elucidating the intricate pathways we have navigated, thereby bringing the big picture into focus. This chapter intertwines the pivotal findings, celebrates the seminal contributions to the existing body of knowledge, illustrates the pragmatic real-world implications, and sketches the tantalizing contours of prospective research landscapes.

This research journey, punctuated by intellectual curiosity and academic rigor, has dared to delve into the crux of network management challenges in the rapidly digitizing world. It has not merely skidded the surface, but has plunged into the depths, challenging orthodox paradigms, and emerging with a unique, trailblazing perspective that could revolutionize how we comprehend and manage networks. The convoluted paths were trodden with a steadfast determination to uncover the myriad potentials of Multi-Agent Systems (MAS) integrated with advanced AI/ML algorithms and softwarized network architectures. The reverberations of these steps have not only resonated within the academic realm but are poised to catalyze significant changes in the practical spheres of network management and telecommunications.

As we bring this chapter and the greater thesis to a close, we do so with the firm conviction that this study is not a terminus but a stepping stone. It provides fertile ground for further exploration, continuing the relentless pursuit of knowledge, understanding, and innovation in the compelling and transformative field of intelligent network management. Therefore, this concluding chapter should be viewed as a reflective consolidation
of the significant achievements and a powerful springboard propelling us into the captivating expanse of future explorations and advancements.

7.2 Summary of Research Findings

This section serves as a beacon, illuminating the crucial conclusions gleaned from our comprehensive exploration in this dissertation. The cornerstone of this work was the envisioning and introduction of a paradigm-shifting modus operandi for network management, one which harnessed the prodigious potential of Multi-Agent Systems (MAS), the computational prowess of Artificial Intelligence/Machine Learning (AI/ML) algorithms, and the fluidity of softwarized network architectures.

As the research journey unfurled, it shone a spotlight on the transformative role of MAS, revealing their latent ability to imbue the network infrastructure with a spirit of autonomy and decision-making prowess, typically associated with holistic entities. This revelation marks a decisive turn in our understanding of network management, catapulting it from the realm of mechanistic processes to that of sentient, responsive, and proactive systems.

The intertwined play of MAS and AI/ML further augments this narrative. It engineers an evolutionary leap, giving birth to a dynamic, adaptable, and self-directed network environment. This emergent ecosystem, underpinned by a synergy of MAS and AI/ML, exhibits an innate capability to thrive amidst the chaos and complexity of the burgeoning digital landscape.

Our research has demonstrated that the sophistication of these intelligent agents when paired with advanced machine learning techniques and the pliability of softwarized networks, can orchestrate a symphony of autonomous decision-making processes. These manifest as self-rectifying, self-configuring, self-administrating, and self-securing actions, culminating in a robust, flexible, and efficient network system.

The fruition of these findings marks a significant milestone, heralding the advent of
a new era of network management. This era is defined by an unprecedented level of autonomy and responsiveness, optimally tuned to manage the ceaseless wave of digital proliferation. Therefore, this section does not merely recapitulate the findings; it heralds the transformation these discoveries signal - a future of network management that is as intelligent and dynamic as the digital ecosystem it governs.

### 7.3 Contributions to Knowledge

In this subsection, we reflect upon the remarkable contributions to our collective knowledge that this thesis has engendered, pioneering advancements that resound with significance in the realm of network management.

Unveiling the multi-faceted potential of Multi-Agent Systems (MAS) in network management has been a cornerstone achievement of this dissertation. Our exploration brought to the fore their inherent capabilities for autonomous and decentralized decision-making, highlighting their capacity to redefine the traditional boundaries of network operations. We bridged the chasm between theory and practice, showcasing the efficacy of MAS within softwarized network architectures, thereby catalyzing a shift from classical to intelligent network governance.

The study further amplified our understanding of AI/ML algorithms within network systems, illuminating the unprecedented value they add in terms of adaptability, efficiency, and robustness. The identification and implementation of specific algorithms like Speed Optimized LSTM and Reinforcement Learning, particularly in the context of predictive network congestion and optimal routing, exemplify how advanced computational intelligence can effectively address contemporary network challenges. This work underscores the fusion of AI/ML techniques with MAS as a catalyst for the birth of networks characterized by inherent smartness and resilience.

Another substantial contribution of this thesis lies in proposing the Scalable and Efficient DevOps (SE-DO) approach. Our study demonstrated how this methodology, woven
into the fabric of the MAS framework, could offer both reactive responses and proactive adaptations to dynamic network requirements. The SE-DO approach, as illustrated, can be instrumental in developing and deploying highly efficient and reliable next-generation networks such as 6G, thereby standing at the forefront of technological evolution.

This research not only ventures into uncharted territories but carves out new pathways in the academic landscape, pioneering an astute integration of Deep Learning paradigms with network link metrics to fundamentally transform digital communication networks, especially within academia. Grounded in a unique dataset of historical user behaviors, this scholarly endeavor intricately uncovers network traffic complexities, revealing bottlenecks and setting the stage for a groundbreaking system proposition. Critiquing the conventional OSPF protocol, this work astutely identifies and rectifies its inherent limitations. In answer to these challenges, we introduce an avant-garde system that seamlessly blends Artificial Intelligence (AI) with Software-Defined Networking (SDN), giving birth to an ultra-proactive QoS agent characterized by its foresight in congestion management and adaptivity in routing. Empirical evidence robustly underpins the novel system’s supremacy over traditional OSPF across pivotal metrics, cementing its transformative potential.

The magnum opus of this research lies also in the introduction and meticulous exploration of the "Adaptive Generalized Proportional Fair (AGPF) scheduler." This scheduler, emerging as the zenith of innovation, harmoniously fuses peak throughput with egalitarian user access, delineating the untapped potential and sheer prowess of adaptive protocols within the wireless domain. Beyond mere elucidation, this work elevates the discourse on wireless communication by showcasing the transformative potential of state-of-the-art packet scheduling algorithms. EmbODYING the very ethos of adaptability, fairness, and performance, these innovative methodologies, meticulously chronicled, stand poised to revolutionize the narrative of wireless communication, all within the overarching architecture of Multi-Agent Systems and Autonomous Network Systems.
The successful design and implementation of an Autonomous, Intelligent, and Advanced AI/ML Framework for Proactive Management and Dynamic Optimal Routing is a testament to the applicability and practical potential of our research findings. This integrated framework, underpinned by continuous learning and adaptation, mirrors the ever-evolving character of 6G networks, thereby meeting the rigorous demands for ultra-low latency, superior reliability, and heterogeneity management.

To summarize, the work encompassed in this thesis has engineered a profound shift in perspective, dismantling conventional notions of network management, and replacing them with a vision that is in sync with the progressive digital zeitgeist. Our research has fostered a crucial paradigm shift in network operations, from mechanistic to intelligent, responsive, and autonomous systems. Consequently, it will inform and inspire both future academic explorations and practical implementations, thereby leaving a lasting imprint on the annals of network management.

7.4 Practical Implications and Industry Relevance

This section underscores the practical reverberations of our research, highlighting its transformative potential to revolutionize the real-world dimensions of network management, architecture, and beyond.

The efficacy of our novel Scalable and Efficient DevOps (SE-DO) approach has substantial real-world implications. It offers a roadmap for optimizing the performance of intelligent agents within resource-constrained environments, equipping networks to adapt to rapidly changing circumstances proactively. By infusing dynamic adaptability into network systems, we can dramatically enhance their performance and resilience, particularly in the context of future networks such as 6G. The SE-DO approach is more than an academic contribution; it provides a practical blueprint for efficient and flexible network management that resonates with the evolving realities of the digital age.

The confluence of Agents and Machine Learning in tasks pertaining to Quality of
Service (QoS) and Radio Resource Management (RRM) using multi-agent reinforcement learning represents a substantial advancement in the telecommunication sector. Our research’s practical translation promises to streamline network operations, significantly augmenting QoS and efficiently managing radio resources. The intelligent QoS agent, as developed in this thesis, offers a way to proactively respond to network traffic and workload distribution, enabling the allocation of the most optimal service path. This breakthrough holds immense potential for network service providers aiming to deliver seamless, uninterrupted service to their end-users.

Our AI/ML Framework for Proactive Management and Dynamic Optimal Routing further demonstrates how our research can revolutionize real-world network management. By accurately predicting network congestion and enabling preemptive action, we can drastically reduce network bottlenecks, ensuring superior end-user experience. Furthermore, by optimizing routing through reinforcement learning, we can maintain high network performance despite dynamically changing network conditions.

In essence, our research impacts the practical sphere of network management profoundly, propelling it towards a future where networks are smart, autonomous, efficient, and resilient. The benefits of this transition are manifold: enhanced user experience, increased reliability, reduced network congestion, and the creation of a more robust, adaptable digital infrastructure. Consequently, our research findings are poised to make a substantial contribution to the ever-evolving telecommunication industry, helping it to embrace the challenges and opportunities of the digital era with confidence and vigor.

7.5 Reflection on the Challenges, Limitations, and Paths to Overcoming

Every intellectual expedition encounters obstacles, and this research is no exception. While we have made substantial strides in paving the way for an autonomous, intelligent, and adaptable network system, we acknowledge the presence of constraints and challenges inherent in our journey. Recognizing these hurdles is not a concession of defeat but rather
a beacon guiding the future trajectory of our research.

The computational cost associated with the implementation of the Multi-Agent Systems, coupled with advanced AI/ML algorithms, is a significant challenge. The performance optimization in resource-constrained environments that our research advocates is a substantial step towards addressing this issue. Still, it does not entirely negate the computational demands. However, with the continued advancement in computing technologies and the evolution of more efficient algorithms, we are optimistic about the possibility of reducing these costs without compromising the system’s performance.

Algorithmic complexity also emerged as a recurrent challenge throughout this research. While complex algorithms can offer nuanced and sophisticated solutions, they can also present difficulties in terms of implementation, interpretation, and troubleshooting. The challenge lies in balancing the sophistication of our algorithms with their practical implementability. As we continue to refine our approach, we remain cognizant of this challenge, aiming for a balance between intricacy and implementability.

The feasibility of deploying our proposed systems within existing network architectures is another area of concern. This challenge underscores the tension between innovation and compatibility, pushing us to create solutions that are not only revolutionary but also adaptable to current infrastructures. We believe in the power of progressive implementation, integrating our systems into existing networks gradually and strategically. We foresee a future where our proposed systems coexist and synergize with existing architectures, leading to a seamless transition towards a more intelligent, autonomous, and adaptable network ecosystem.

In summary, while these challenges and limitations present real obstacles, they are surmountable. They provide valuable insights into the areas requiring further exploration and optimization. Importantly, they enrich our research’s narrative, acting as catalysts for further innovation, refinement, and progress in the exciting field of intelligent network management.
7.6 Blueprint for Future Exploration: Recommendations and Aspirations

The realm of intelligent network management, in its dynamic flux and constant evolution, presents a fertile landscape for continued exploration and innovation. It is a domain where the boundaries of the known are perennially pushed, offering an expansive playground for future endeavors. This research, while a significant step forward, is by no means the terminus of our intellectual journey. As such, we propose several directions for future work.

Addressing the limitations and challenges identified in this research is of paramount importance. There lies an immediate need to further investigate methods to lower computational costs, streamline algorithmic complexity, and devise ways to better integrate our proposed systems with existing network architectures. These form a trifold priority for future research endeavors and pave the way towards a more efficient, implementable, and compatible intelligent network management system.

In terms of specific algorithmic improvements, our research has shown that reinforcement learning offers promising results for Quality of Service and Radio Resource Management tasks. However, the field of reinforcement learning is vast, and our exploration has only skimmed the surface. A plethora of reinforcement learning algorithms, each with unique strengths and characteristics, await examination. Future research can expand on this, scrutinizing diverse algorithms to ascertain which can provide even greater efficiency and optimization for these tasks.

Moreover, we have primarily situated our research within the context of 5G, B5G, and 6G networks. But as the wheel of technological innovation turns relentlessly, the dawn of the 7G era is on the horizon. This upcoming generation of networks will inevitably come with its own unique requirements and challenges. A pertinent area of future research would be to explore how the Multi-Agent Systems approach can be evolved and adapted to these novel architectures. It is imperative to prepare for this future today and work
proactively to ensure that our systems remain at the cutting edge of technology, maintaining their relevance and applicability.

In this vast expanse of intelligent network management, the path we have traversed in this research is merely the beginning, opening new frontiers of exploration. This dissertation, while substantial in its contributions, only marks the initiation of a journey destined to evolve continually. Recognizing the dynamic nature of the field, we propose the following avenues for future research and development:

- **Groundwork for Practical Implementation and Evaluation:** The ultimate litmus test for our Multi-Agent-based Network Automation of the Network Management System (MANA-NMS) would be real-world implementation and evaluation. An essential step forward would be the practical deployment of MANA-NMS within a live network environment, using the Python Agent Development Environment (PADE) framework or a similar framework. Such a hands-on application would give invaluable insights into the system’s performance, scalability, and reliability, identifying gaps and setting the stage for enhancements.

- **Augmenting Security and Resilience:** In a world increasingly susceptible to cyber threats, the fortification of security and resilience within the multi-agent system architecture demands priority. Further research can explore techniques for integrating secure communication protocols, authentication mechanisms, intrusion detection systems, and anomaly detection algorithms. These would bolster the system’s defensive capabilities, ensuring the continuous, secure, and unassailable operation of our networks.

- **Advancements in Resource Allocation:** The quest for optimal resource allocation continues, especially in the context of resource-constrained environments. Future work can delve deeper into this area, refining the Scalable and Efficient DevOps (SE-DO) approach and exploring advanced optimization algorithms to achieve
maximum utilization of network resources, keeping in mind the ever-fluctuating network traffic, workload distribution, and dynamic requirements.

- **Reinforcement Learning Expansions for Network Management:** While our research has exploited reinforcement learning for Quality of Service (QoS) and radio resource management (RRM), its application to other network management tasks remains largely untapped. Future research can extend the use of reinforcement learning to areas such as fault detection and recovery, network optimization, and energy efficiency, thus broadening the horizons of intelligent network management.

- **Integration with Cutting-Edge Technologies:** Our proposed multi-agent approach could synergize with emerging technologies like edge computing, blockchain, and Internet of Things (IoT) to catalyze the evolution of intelligent network management. This integration could enhance the capabilities of the multi-agent system, enabling it to navigate complex and distributed environments with finesse and acumen.

- **Strides in Standardization and Interoperability:** Another compelling area of future research lies in the standardization and interoperability of our proposed approach. Investigating how multi-agent systems can integrate seamlessly with existing network architectures and frameworks, ensuring compatibility, and establishing best practices for the deployment of autonomous network systems could pave the way for more harmonized, unified networks.

- **Security in Agent-Based Networks:** An increasingly connected world opens up new vulnerabilities that cybercriminals are eager to exploit. Therefore, exploring the security aspects of agent-based networks, focusing on potential weaknesses, threats, and countermeasures, would be invaluable. Techniques that ensure secure operation of intelligent network management systems and safeguard against cyber threats in agent-based architectures would significantly bolster network integrity.
• **Privacy and Trust in Autonomous Networks:** With increasing autonomy comes increasing scrutiny towards privacy and trust. Developing mechanisms to enhance user privacy, protect sensitive data, and establish trust among agents in a distributed and autonomous environment would be crucial. Here, exploration into privacy-preserving techniques, secure communication protocols, and decentralized trust models could yield significant insights.

• **Integration with Space Cybersecurity:** As we reach for the stars, our network systems need to keep pace. Exploring the integration of the multi-agent approach with space cybersecurity measures, aiming to enhance the cybersecurity of space-based systems, can be a monumental stride. By navigating the unique challenges and requirements of space cyber defense with agent-based architectures, we can fortify our cosmic communications and operations.

• **Resilience and Fault Tolerance:** The inherent dynamism and unpredictability of network systems necessitate resilience and fault tolerance. Future work can focus on techniques for fault detection, recovery, and self-healing mechanisms in the face of failures or attacks, thereby ensuring the continuity and high availability of network services.

• **Scalability and Performance Optimization:** As our digital world expands, so too must our networks. Investigating approaches to improve scalability and performance in large-scale deployments, managing numerous agents, distributing computational tasks, and optimizing resource utilization are areas ripe for exploration.

• **Interoperability and Standardization:** To create harmonious networks that seamlessly integrate and collaborate, addressing interoperability and standardization challenges is critical. Developing protocols and frameworks to enable this
collaboration, along with exploring standardization efforts and guidelines, can promote the adoption and deployment of autonomous network systems.

This research, rich in its significant advancements, serves not as a destination, but as a profound launching pad for untapped exploration in the ever-evolving field of intelligent network management. It signifies the commencement of an exciting journey, guided by the relentless spirit of curiosity, innovation, and progress, into the enticing arena of endless opportunities and uncharted territories. The future, a vast expanse of infinite possibilities, lies before us, and this research acts as a potent catalyst, propelling us into this expansive realm with unflagging commitment. As we navigate the long, winding path ahead, every stride we take, every challenge we surmount, is a step towards a more connected, intelligent world. We look ahead with eager anticipation, ready to embrace what lies beyond the horizon in our quest for furthering the frontiers of knowledge. We are, indeed, excited to see where this extraordinary path takes us and the remarkable advancements it unfurls along the way.

7.7 Concluding Remarks

As we sail further into the digital age, the necessity for a paradigm shift from conventional network management methodologies to increasingly autonomous and intelligent systems has become paramount. The exploration marshalled through this thesis has laid a robust foundation in the revolutionary transformation of intelligent network management, shedding significant light on the remarkable potential and practicality of Multi-Agent Systems (MAS) and Artificial Intelligence/Machine Learning (AI/ML) in heralding this change.

Through rigorous analysis, comprehensive experimentation, and innovative application, we have ventured into the profound depths of AI and MAS, effectively pushing the frontiers of our understanding and implementation. In this endeavor, we have unearthed
significant insights and developed novel methodologies, all of which have served to accelerate our journey towards a future where networks evolve beyond their traditional roles.

The networks of the future, as we envisage, will transcend the simple role of serving as channels for information. They will morph into sophisticated, intelligent entities, capable of dynamic adaptation and autonomous decision-making. Leveraging the power of MAS and AI/ML, they will proactively navigate the ever-changing digital landscape, dynamically adapting to evolving requirements, anticipating challenges, and making informed decisions with minimal human intervention.

We stand at the precipice of a new era where these intelligent systems will redefine the way we understand and interact with our digital world. This research, while a pivotal contribution to this field, is just the beginning of this transformational journey. We remain cognizant of the vast, unchartered territories that lie ahead and the challenges that they present. Yet, we are optimistic and eager to delve deeper, spurred on by our relentless pursuit of innovation and progress.

In retrospect, this thesis presents not merely a change in perspective, but a radical redefinition of possibilities. It is an invitation to perceive and shape the future of our interconnected digital world through a lens of transformative innovation and outstanding progress. As we conclude, we do so with the awareness that the journey has only just begun. With unflagging determination, we look towards the horizon, ready to explore the unexplored, to answer the unanswered, and to unlock the future of intelligent network management in next-generation networks.
### Time Complexity Analysis of the Training Agent Class

<table>
<thead>
<tr>
<th>Function</th>
<th>Time Complexity</th>
<th>Explanation</th>
<th>Determining Factor(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>on_start()</code></td>
<td>O(1)</td>
<td>The function is a simple call to the parent class and scheduling a call to <code>training_agent()</code>. It does not depend on the input size.</td>
<td>-</td>
</tr>
<tr>
<td><code>training_agent()</code></td>
<td>O(N + NLogN + K + M)</td>
<td>This function includes preprocessing the data (O(N)), computing class weights (O(N)), splitting the dataset (O(NLogN)), training the model (O(K) where K is dependent on the number of epochs, batch size, and model complexity), and saving and plotting the model (O(M), mainly depends on the number of data points).</td>
<td>Size of the dataset (N), Model complexity and training configuration (K), Number of data points for saving and plotting the model (M)</td>
</tr>
</tbody>
</table>
### Time Complexity Analysis of the Efficient Multi-Objective Optimization Algorithm for Network Data Collection

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
<th>Time Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initialization (line 3)</td>
<td>Assuming initialization of variables takes constant time.</td>
<td>$O(1)$</td>
</tr>
<tr>
<td>Assign weights (line 7)</td>
<td>Since the number of metrics seems to be a small, fixed number.</td>
<td>$O(1)$</td>
</tr>
<tr>
<td>Define multi-objective function (line 9)</td>
<td>Assuming that the function takes constant time.</td>
<td>$O(1)$</td>
</tr>
<tr>
<td>Define constraints for optimization (line 11)</td>
<td>The complexity depends on the function’s implementation, assuming the function is constant time.</td>
<td>$O(1)$</td>
</tr>
<tr>
<td>Select suitable optimization algorithm (line 13)</td>
<td>The complexity depends on the method used to select the optimization algorithm, assuming a fixed set of algorithms to choose from.</td>
<td>$O(1)$</td>
</tr>
<tr>
<td>Minimize objective function with constraints (line 15)</td>
<td>The complexity depends on the optimization algorithm selected. For example, with a genetic algorithm, the complexity is $O(\text{GEN*POP})$.</td>
<td>$O(\text{GEN*POP})$</td>
</tr>
<tr>
<td>Evaluate performance of $X_{\text{opt}}$(line 17)</td>
<td>The complexity depends on the specifics of the EvaluatePerformance function, assuming it needs to iterate over all elements in $X_{\text{opt}}$.</td>
<td>$O(N)$</td>
</tr>
<tr>
<td>Visualize results (line 19)</td>
<td>The complexity depends on the specifics of the visualization function. For instance, if it involves sorting, it could be $O(N \log N)$.</td>
<td>$O(N \log N)$ or function-specific</td>
</tr>
</tbody>
</table>
## Time Complexity Analysis of the Pioneering SP-LSTM Framework for Anticipating Network Bottlenecks

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
<th>Time Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initialization (line 2)</td>
<td>Assuming initialization of variables takes constant time.</td>
<td>$O(1)$</td>
</tr>
<tr>
<td>For each element in sequence (lines 3-12)</td>
<td>These operations are done for each element in the sequence. Matrix operations have a complexity of $O(F^2)$ and sequence operations have a complexity of $O(T)$.</td>
<td>$O(T \times (T + F^2))$</td>
</tr>
<tr>
<td>Output (line 13)</td>
<td>The complexity depends on whether the full sequence or only the final state is returned, but in either case, it’s at most $O(T)$.</td>
<td>$O(T)$</td>
</tr>
</tbody>
</table>
Appendix B

Key Features of Raspberry Pi 4 Model B

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processor</td>
<td>Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz</td>
</tr>
<tr>
<td>Memory</td>
<td>8GB LPDDR4</td>
</tr>
<tr>
<td>Connectivity</td>
<td>2.4 GHz and 5.0 GHz IEEE 802.11ac wireless, Bluetooth 5.0, BLE; Gigabit Ethernet</td>
</tr>
<tr>
<td>USB Ports</td>
<td>2 USB 3.0 ports; 2 USB 2.0 ports</td>
</tr>
<tr>
<td>GPIO</td>
<td>Standard 40-pin GPIO header (fully backward-compatible with previous boards)</td>
</tr>
<tr>
<td>Video</td>
<td>2 micro HDMI ports (up to 4kp60 supported)</td>
</tr>
<tr>
<td>Audio</td>
<td>2-lane MIPI DSI display port, 2-lane MIPI CSI camera port, 4-pole stereo audio and composite video port</td>
</tr>
<tr>
<td>Storage</td>
<td>Micro-SD card slot for loading operating system and data storage</td>
</tr>
<tr>
<td>Power</td>
<td>USB-C connector for main power supply, 5V DC via GPIO header</td>
</tr>
<tr>
<td>Operating Temperature</td>
<td>0 – 50 degrees C ambient</td>
</tr>
</tbody>
</table>

The Raspberry Pi, a brainchild of the Raspberry Pi Foundation, is a compact single-board computer designed to stimulate hands-on exploration in computer science within academic contexts. Its charm lies in its compatibility with an array of operating systems, most notably the Raspbian OS, Ubuntu, Windows 10 IOT Core, and RISC OS.
Encouraging the use of Python and Scratch as its primary programming languages, it nonetheless supports a diverse range of other languages.

Initiating the Raspberry Pi involves installing the desired operating system and attributing a static IP to the unit. This process guarantees a consistent IP address, streamlining the networking aspect. Upon completion of the Raspberry Pi's setup, it is possible to fine-tune its network settings to enable smooth interfacing with the NetSim Emulator.

The fusion of the Raspberry Pi with NetSim, a potent network simulation tool, necessitates a physical connection between the sensor and Raspberry Pi. Subsequently, the Raspberry Pi is integrated into NetSim's virtual network via an emulation application, mapping the Raspberry Pi to a specific virtual node.

Raspberry Pi 4 Model B
Appendix C

Agent-Based Resource Allocation Simulation Script

Overview:
The simulation script evaluates various agent-driven resource allocation strategies (PF, GPF, and AGPF) within telecommunication environments, harnessing the robustness of the Sacred library for experiment tracking and the OpenAI Gym framework for environment setup.

Configuration:

- **Agent Parameters**: Sourced from `config.agent.json`, it defines the behavior and characteristics of the agent.

- **Experiment Parameters**: Extracted from `config.sacred.json`, this encompasses settings related to Sacred experiment tracking.

- **Environment Parameters**: Detailed in `config.environment.json`, they set the context and rules for the simulation.

Experiment Initialization:

- Using the agent type from the configuration, the experiment is configured and augmented with additional parameters.

- A MongoDB connection string is crafted to facilitate the potential storage of results in a MongoDB database, though the actual storage feature remains commented out.

Simulation Mechanics:

- A core function, `main`, decorated with `@ex.automain`, runs the heart of the simulation.
• For each episode, a relevant environment (either for Time-Frequency or NOMA UL Time-Frequency resource allocation) is initialized.

• Agents, ranging from strategies like "random" to more sophisticated ones like "proportional fair”, are instantiated based on configurations.

• Within each episode, agents make decisions based on the environment’s state, with results logged periodically, providing insights into metrics like CQI, buffer occupancy, and QoS Identifier.

Post-Simulation Analysis:

• Concluding the simulation episodes, mean rewards across episodes are logged, offering an aggregate view of agent performance.

Remarks:
The script emphasizes modularity, allowing seamless adjustments via configuration files. This design ensures that the platform remains flexible for exploring a plethora of agent strategies in diverse telecommunication settings. It’s important to acknowledge that we have adapted and built upon the foundational scripts available at Nokia’s Wireless Suite on GitHub. This source provided a valuable starting point for our advanced customizations and enhancements to cater to our specific research objectives.

In our exploration of adaptive resource allocation strategies, we developed a specialized agent named the Adaptive Generalized Proportional Fair Channel Aware Agent (AGPF). This agent leverages the principles of Proportional Fairness but incorporates channel-aware adaptability to better cater to varying telecommunication environments. A snapshot of the core definition and logic behind the AGPF is illustrated in the following code:
class AdaptiveProportionalFairChannelAwareAgent(ProportionalFairChannelAwareAgent):
    CQI2SE = [..................]
    def __init__(self, action_space, n_ues, buffer_max_size, alpha, alpha_min=0.2, alpha_max=0.5, buffer_size_threshold =0.5):
        super().__init__(action_space, n_ues, buffer_max_size, alpha, alpha_min=0.001, alpha_max=0.1, buffer_size_threshold=0.2)
        self.alpha = alpha
        self.alpha_min = alpha_min
        self.alpha_max = alpha_max
        self.buffer_size_threshold = buffer_size_threshold
    def _calculate_priorities(self, cqi, o, b, buffer_size_per_ue):
        se = np.zeros(shape=(self.K,))
        for i in range(16):
            se[cqi == i] = self.CQI2SE[i]

        self.alpha = self.alpha_min + (self.alpha_max - self.alpha_min) * np.exp(-self.buffer_size_threshold * buffer_size_per_ue.mean())

        priorities = (1 + o) / (b ** self.alpha) * buffer_size_per_ue * se
        return priorities

Adaptive Proportional Fair Channel Aware Agent Class
References


[40] F Keras Chollet. Available online: https://keras.io (accessed on 22 april 2023). © 2019 by the authors. *Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/), 2015.*


