

Research Project Proposal

NLP-Driven Intent Based Networking Layered-Architecture for Software Defined Networks

G12

E/19/409
Dineth Udugamasooriya

E/19/413
Nipul Viduranga

E/19/446
Kalindu Wijerathna

Department of Computer Engineering

Faculty of Engineering

University of Peradeniya

2025

Research Project Proposal

NLP-Driven Intent Based Networking Layered-Architecture for Software Defined Networks

G12

E/19/409
Dineth Udugamasooriya



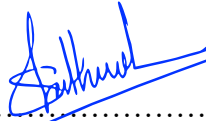
E/19/413
Nipul Viduranga



E/19/446
Kalindu Wijerathna



Supervised By



Dr. Suneth Namal

Department of Computer Engineering

Faculty of Engineering

University of Peradeniy

Abstract

Software-Defined Networking (SDN) has revolutionized network management by decoupling the control and data planes, enabling dynamic and programmable network architectures. Building on this foundation, Intent-Based Networking (IBN) enables administrators to express high-level business policies in natural language, which are then automatically translated into network configurations. This project proposes an innovative NLP-driven intent-based networking layered architecture for SDNs, aiming to streamline the translation of human-readable intents into machine-executable configurations. By integrating advanced Natural Language Processing (NLP) techniques with machine learning (ML) methods, the architecture addresses critical challenges such as accurate intent translation, intent conflict resolution, and dynamic optimization of network policies through threshold tuning. By evaluating in a controlled SDN environment, the proposed framework seeks to enhance network management efficiency, improve operational performance, and pave the way for more autonomous networking systems.

Table of Contents

Chapter 1 : Introduction	1
1.1 Introduction.....	1
1.2 Aims and Objectives	2
1.2.1 Aim.....	2
1.2.2 Objectives	2
1.3 Structure of the Dissertation	3
Chapter 2 : Advancements in NLP-Driven Intent-Based Networking for Software- Defined Networks	4
2.1 Introduction.....	4
2.2 Intent-Based Networking Architectures and Frameworks	4
2.2.1 Foundational SDN-Based IBN Frameworks	5
2.2.2 Domain Specific IBN Frameworks.....	6
2.2.3 AI & NLP-Enhanced IBN frameworks.....	7
2.2.4 Advanced programmable & policy-driven IBN frameworks.....	8
2.3 NLP-Driven Intent Processing in SDN.....	11
2.4 Intent Conflict Resolution.....	14
2.5 Machine Learning for SDN Optimization	16
2.6 Intent Based Network Monitoring and Telemetry	18
2.7 Summary	20
Chapter 3 : Reshaping SDN: Harnessing NLP and ML to Empower Intent-Based Networking	22
3.1 Introduction.....	22
3.2 Natural Language Processing (NLP) for Intent Understanding, Named Entity Recognition and Policy Generation	22
3.3 Machine Learning for Intent Optimization	23
3.4 Evaluating NLP-Driven Intent-Based Networking Through SDN Emulation ..	24

3.5 Summary	24
Chapter 4 : An NLP-Driven Approach to Intent Interpretation for SDN Configuration	25
4.1 Introduction.....	25
4.2 Intent Processing Flow.....	25
4.3 SDN Execution Flow	28
4.4 Evaluation	30
4.5 Summary	31
Chapter 5 : Analysis and Design.....	32
5.1 Introduction.....	32
5.2 System Overview Diagram	32
.....	33
5.3 Module Breakdown.....	34
5.4 Summary	36
Chapter 6 : Implementation	37
6.1 Introduction.....	37
6.2 Emulation Setup Description and Topologies.....	37
6.2.1 Proposed Key Technologies and Justifications	37
6.2.2 High-Level Implementation Plan.....	39
6.2.3 Proposed Topologies	41
6.3 Summary	44
Chapter 7 : Discussion	46
7.1 Introduction.....	46
7.2 Comparison with Existing Work.....	46
7.3 Research Direction.....	47
7.3.1 Immediate Impact and Emerging Trends in IBN and SDN	47
7.3.2 Long-Term Impact on Network Automation and Intelligence.....	48

7.3.3 Potential Challenges and Research Areas for the Next Decade.....	49
7.3.4 What to Expect in the Coming Decades	49
7.3.5 Conclusion	50
7.4 Summary	50
References.....	52
Appendix A	56
Appendix B	57
Appendix C	58

List of Figures

Figure 2.1: Overview of Related Works in NLP-driven Intent-Based Networking.....	4
Figure 2.2: Overview of the Categorization of Selected Prominent IBN Architectures/ Frameworks.....	5
Figure 4.1: Intent Processing Flow.	26
Figure 4.2: SDN Execution Flow.....	28
Figure 5.1: High Level Design of the Proposed NLP-Driven IBN Layered Architecture.....	33
Figure 6.1: High Level Implement-ation Plan	39
Figure 6.2: Proposed Collapsed Core Topology	42
Figure 6.3: Proposed Spine-Leaf Topology	43
Figure 6.4: Proposed Fat-Tree Topology	44

List of Tables

Table 2.1: Comparative Analysis of Prominent IBN Frameworks and Architectures.	10
Table 2.2: Comparative Analysis of NLP-Based Intent Processing Approaches.....	13
Table 2.3: Summary of DRL-Based Routing Approaches.....	17

Chapter 1 : Introduction

1.1 Introduction

Software-Defined Networking (SDN) has revolutionized network management by separating the control plane from the data plane, enabling more flexible and programmable network architectures. One of the most promising developments within SDN is Intent-Based Networking (IBN), which allows network administrators to define high-level business policies and objectives (referred to as intents) while abstracting away the complex network configurations required to achieve them. Automation, efficiency, and network management have all significantly improved as a result of this move towards intent-driven approaches.

The role of Natural Language Processing (NLP) in IBN has gained substantial attention due to its potential to bridge the gap between human-readable intent specifications and machine-executable network configurations. NLP-driven approaches allow network administrators to express complex network behaviors in natural language, which can then be interpreted and translated into actionable network configurations. Recent advancements have demonstrated the potential of NLP to automate intent translation, conflict resolution, and network optimization, particularly in Software-Defined Networks (SDNs).

As the field continues to mature, the integration of AI and machine learning (ML) techniques is expected to further enhance IBN systems, enabling predictive analytics, self optimization, and improved network performance, this research proposal aims to introduce a novel framework that harnesses the power of NLP-driven intent processing to enhance SDN operations. By exploring the current state of NLP applications in IBN, highlighting key technological advancements, and addressing existing challenges, this work seeks to lay the groundwork for developing a novel NLP driven intent based networking layered architecture for SDNs.

1.2 Aims and Objectives

1.2.1 Aim

To develop an innovative NLP-driven intent-based networking framework for Software-Defined Networks (SDNs) that simplifies network management by transforming high-level user intents into dynamic, machine-executable network configurations.

1.2.2 Objectives

1. Framework Design and Architecture

Develop a conceptual, layered architecture that integrates NLP-driven intent processing with SDN control mechanisms.

2. NLP Module Development

Design and implement NLP algorithms capable of parsing and understanding high-level natural language network intents and create translation modules that accurately map interpreted intents to specific network policies and configurations.

3. Dataset Collection and Annotation

Gather a comprehensive dataset comprising diverse natural language network intents paired with their corresponding network configurations and implement data cleaning and preprocessing steps to prepare the dataset for training and evaluation.

4. Machine Learning Integration for Optimization

Explore and implement ML techniques to optimize the threshold values of generated network policies.

5. Prototype Implementation

Build a working prototype of the proposed framework within a controlled SDN environment or simulation platform.

6. Performance Evaluation and Validation

Establish evaluation metrics such as translation accuracy, configuration latency, and overall network performance improvements and conduct experiments under various network conditions and workloads to assess the system's effectiveness and reliability.

1.3 Structure of the Dissertation

Chapter 1: Introduction introduces the research topic, laying the foundation by presenting the aims and objectives of the study. The chapter defines the overall aim and outlines the specific objectives to be achieved. In Chapter 2: Advancements in NLP-Driven Intent-Based Networking for Software-Defined Networks, the focus shifts to the evolution of IBN frameworks within SDN, covering foundational, domain-specific, and AI-enhanced frameworks. It also explores advanced programmable and policy-driven IBN frameworks, as well as NLP-driven intent processing, conflict resolution, and machine learning for SDN optimization. Chapter 3: Reshaping SDN: Harnessing NLP and ML to Empower Intent-Based Networking delves into the practical use of NLP for intent understanding, named entity recognition, and policy generation. It also explores machine learning for intent optimization and evaluates NLP-driven IBN through SDN emulation. Moving into Chapter 4: An NLP-Driven Approach to Intent Interpretation for SDN Configuration, this chapter presents the process flow for intent interpretation and SDN execution, followed by a detailed evaluation of the approach's effectiveness. Chapter 5: Analysis and Design provides an overview of the system design, breaking down the modules involved and presenting a system overview diagram. In Chapter 6: Implementation, the emulation setup and network topologies used for testing the system are described in detail. Finally, Chapter 7: Discussion compares the research findings with existing work, offering insights into the contribution of this study and suggesting potential directions for future research.

Chapter 2 : Advancements in NLP-Driven Intent-Based Networking for Software-Defined Networks

2.1 Introduction

This chapter categorizes and discusses **related works** in different aspects of IBN, including architectural frameworks, NLP-driven intent processing in Software-Defined Networking (SDN), intent conflict resolution, machine learning for SDN optimization, and intent-based network monitoring and telemetry. Each subsection provides an overview of existing research contributions, highlighting key methodologies, challenges, and innovations that shape the current state of IBN development. Figure 2.1 provides an overview of the key areas explored in the literature.

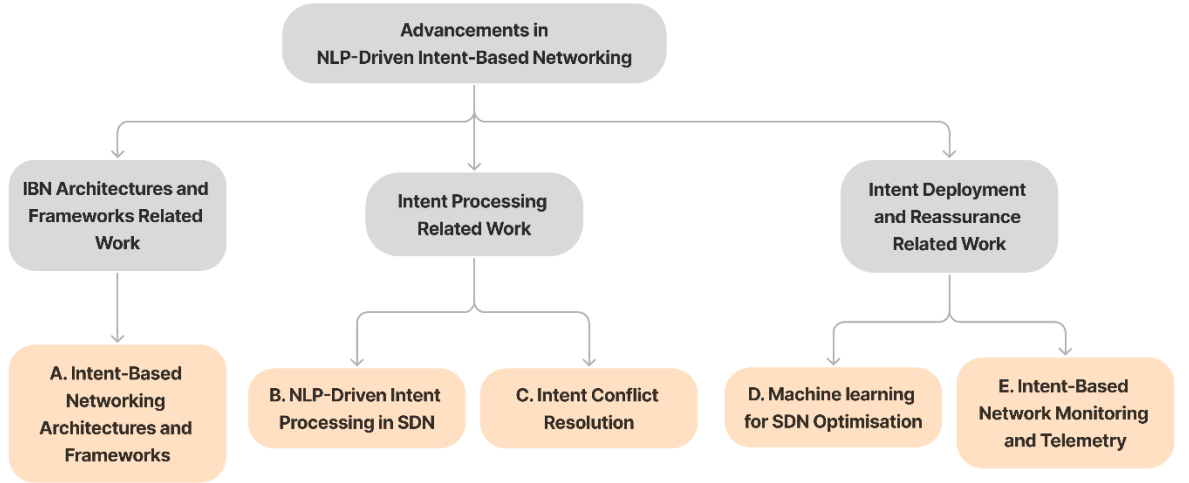


Figure 2.1: Overview of Related Works in NLP-driven Intent-Based Networking.

2.2 Intent-Based Networking Architectures and Frameworks

Over the years, IBN architectures have evolved from fundamental Software-Defined Networking (SDN)-based models to domain-specific solutions, AI-driven automation, and programmable network frameworks. This section categorizes and reviews notable IBN frameworks, illustrating their advancements and identifying key research

challenges. Figure 2.2 provides an overview of the categorized prominent IBN architectures chosen for this study.

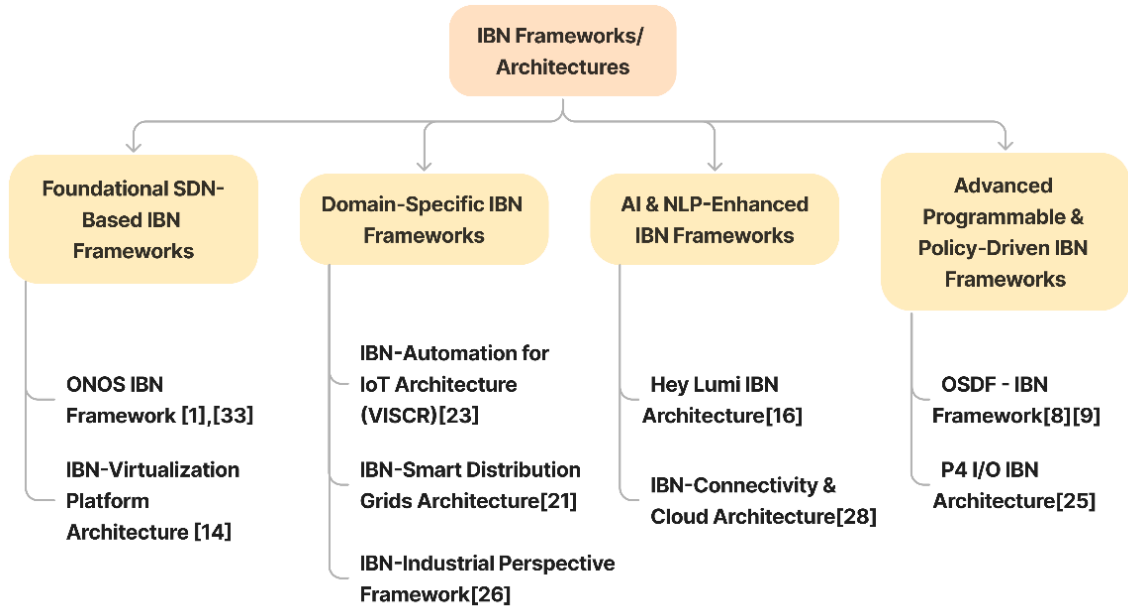


Figure 2.2: Overview of the Categorization of Selected Prominent IBN Architectures/ Frameworks.

2.2.1 Foundational SDN-Based IBN Frameworks

These architectures provide the foundation for IBN frameworks by introducing intent translation, flow rule installation, and basic SDN control operations. The ONOS Intent Framework, one of the first instances of frameworks on Intent Based Networking (IBN), offers administrators a methodical approach to defining high-level "intent." The flow rules that can be put on devices with Software-Defined Networking (SDN) capabilities are then automatically generated from these intentions. Applications can communicate with the SDN controller through the northbound interface (NBI) provided by the ONOS platform, which acts as an SDN controller. These intentions can be defined and sent by administrators using a variety of tools, such as REST APIs, Command Line Interfaces (CLI), or specially designed apps that make use of the ONOS framework. [33] Both the proactive and reactive execution models are supported by the ONOS Intent Framework. By pre-installing flow rules based on anticipated network conditions, proactive execution makes sure the network stays in line with predicted states. Conversely, reactive execution reacts

to changes in the network in real-time, including bandwidth modifications or link failures. Though this flexibility is useful, the framework's current drawbacks include its high reliance on human intent registration and its limited expressiveness for handling complex application requirements. Its usefulness and scalability in expansive, dynamic network systems are greatly impacted by these limitations. According to the study, the framework's reliance on manual intervention, which grows more time consuming and laborious as the network size grows, makes it difficult to manage dynamic, large-scale situations [1].

Extending ONOS, the IBN-Virtualization Platform introduces multi-tenancy support through network slicing and virtualization layers [14]. This architecture maps intent-defined virtual networks (VNs) onto a physical SDN infrastructure, ensuring logical isolation of different tenants. The platform incorporates advanced mechanisms for conflict detection and resolution, which are crucial for maintaining the integrity of network operations across multiple tenants. By normalizing intents and applying precedence rules, the system minimizes the risk of overlapping policies. Despite its scalability advantages, concerns arise as the number of tenant-defined policies increases. The complexity of managing these interactions can lead to performance bottlenecks, particularly in environments with extensive network slicing requirements. This highlights the need for improved mechanisms to handle dynamic policy updates and conflict resolution efficiently.

2.2.2 Domain Specific IBN Frameworks

These frameworks tailor IBN capabilities to specific network environments, such as IoT security, smart grid automation, and industrial networks. IBN-Automation for IoT architecture (VISCR) introduces a vendor-independent policy translation mechanism that uses graph modeling to address security and policy enforcement issues in IoT infrastructures [23]. By abstracting disparate vendor-specific rules into a single framework, this design makes it possible to enforce policies consistently across a variety of devices. Policy administration is made easier by the VISCR system's five-component architecture, which includes Code Sanity & Policy Graph Generation, Conflict Detection & Resolution, and Infrastructure Abstraction Engine. For real-time conflict detection and resolution, the substantial processing overhead is still a

drawback. Improving computational efficiency and creating machine learning methods to anticipate such policy conflicts are necessary because the complexity of analyzing massive volumes of data generated by IoT devices can delay prompt policy enforcement.

IBN-Smart distribution grids architecture allows for low-latency, SLA-compliant automation for power distribution grids by integrating 5G network slicing with IBN. It uses an architecture with three layers [21]. The intent management layer is responsible for translating grid operator intents into service requirements, utilizing Natural Language Processing (NLP) to extract specific needs from operators. Network management layer manages the lifecycle of network slices, allocating resources based on the dynamic demands of the power grid. Grid infrastructure layer integrates 5G Radio Access Networks (RAN), Supervisory Control and Data Acquisition (SCADA) systems, and power infrastructure for real-time monitoring and automation. Even with its scaling benefits, issues with multi-domain coordination and latency still exist. The orchestration of services becomes more difficult when many domains are integrated, particularly when real-time data processing and decision making are involved.

IBN-Industrial perspective framework is designed for enterprise and industrial networks [26]. This framework integrates AI-driven optimization through an Optimization & Decision Maker (ODM) and a Network State Observer (NSO). The ODM formulates optimization problems based on historical intent data, dynamically adjusting network configurations to suit current conditions. The NSO continuously monitors the network state, providing feedback to ensure compliance with operational intents.

2.2.3 AI & NLP-Enhanced IBN frameworks

These frameworks focus on improving intent usability and automating service mapping using natural language processing (NLP) and AI-driven decision-making. Hey Lumi IBN framework pioneers natural language-based intent parsing, allowing network operators to express requirements in human-readable form [16]. By utilizing Named Entity Recognition (NER) and deep learning techniques, Hey Lumi converts high-level user intents into structured policies in the Network Intent Language (Nile).

This transformation simplifies the process of specifying network configurations, making it more accessible to non-technical users. However, ambiguities in natural language interpretation can lead to misconfigurations, and limited enforcement mechanisms restrict its reliability in production environments. Additionally, Hey Lumi lacks the programmability of frameworks like P4 IO [25] and OSDF [8], [9], which allow for direct, low-level control over packet processing and dynamic adaptation of network behaviors. While its high-level abstraction simplifies network management, the inability to define granular, programmable logic limits its applicability in scenarios requiring fine-tuned, real-time optimizations. Enhancing the accuracy of NER models and incorporating fallback mechanisms for ambiguous intents are critical areas for improvement.

Targeting cloud-network integration, IBN-Connectivity & cloud architecture AI-enhanced framework maps high-level service requirements to connectivity models using machine learning-based ranking algorithms [28]. By incorporating cross-domain service modeling, it enables unified intent based management for both cloud and network services. This approach streamlines the provisioning process while ensuring that services meet user-defined SLAs. Despite its potential, challenges remain regarding interoperability across different cloud providers and security vulnerabilities in cross-domain automation. Additionally, this framework also lacks programmability compared to more flexible solutions like P4 IO and OSDF, which allow for deeper customization and real-time optimization of network behaviors. Its reliance on predefined models and machine learning based ranking limits granular control over traffic engineering and dynamic resource allocation. Establishing standardized protocols for service definitions, enhancing programmability for fine-tuned network adjustments, and ensuring compliance with security regulations are essential steps toward improving trust, usability, and adaptability in multi-cloud environments.

2.2.4 Advanced programmable & policy-driven IBN frameworks

These frameworks represent the most advanced IBN solutions, leveraging policy-driven automation and programmable network functions (such as P4-based programmability).

The Open Software Defined Framework (OSDF) introduces policy-driven SDN programming, supporting hybrid intent execution that combines proactive rule installation with reactive adaptation [8], [9]. This approach ensures that network policies are enforced efficiently while allowing for real-time adjustments based on network conditions. Compared to purely reactive approaches, such as those discussed in Reactive Configuration Updating for Intent-Based Networking [29], OSDF reduces the risk of performance degradation caused by frequent, on-the-fly updates. The reactive approach, while beneficial for dynamic environments, often encounters challenges related to update latency and potential policy inconsistencies when multiple network events occur in rapid succession. To maintain consistency and security, OSDF employs advanced policy resolution techniques that identify redundancies, overlaps, and security violations within intent-based configurations. This contrasts with fully reactive models, which primarily focus on timely updates but may lack built-in mechanisms for conflict resolution. Also, OSDF conflict resolution can be analyzed alongside Merlin's [27] approach. Merlin was not considered a complete IBN framework in this study because it is designed as a high-level language for specifying network policies, rather than a full IBN system that encompasses policy translation, enforcement, and lifecycle management. However, Merlin's conflict resolution mechanisms offer valuable improvements for frameworks like OSDF. While OSDF integrates predefined resolution strategies to handle conflicts, it lacks the programmable, dynamic conflict negotiation that is a hallmark of the Merlin language. Merlin's Negotiator model actively resolves policy conflicts by negotiating constraints between network elements and adapting dynamically to network changes and interdependencies. In contrast, OSDF employs a static conflict resolution approach, where predefined rules aim to resolve overlapping intents without dynamic adjustment.

The most sophisticated IBN paradigm is the P4I/O framework, which combines intent-driven automation with P4 programmability. P4I/O allows dynamic intent compilation, which transforms high-level intents into runtime-modifiable P4 programs, in contrast to conventional SDN-based IBN systems [25]. Because of its adapt ability, network configurations may be quickly changed to meet changing needs. To maximize packet processing, P4I/O builds Directed Acyclic Graphs (DAGs) and uses predefined code templates. Stateful pipeline alterations are supported by the

design, allowing for real-time reconfiguration without interfering with traffic that is already flowing. Notwithstanding its benefits, a barrier to entry is the need for specific P4 programming expertise for deployment. Furthermore, there is still room for investigation into AI-driven methods for improving intent processing with reference to this architecture.

Table 2.1: Comparative Analysis of Prominent IBN Frameworks and Architectures.

Framework / Architecture	Intent Specification	Processing Approach	Description	Challenges & Research Gaps
ONOS IBN Framework [1], [33]	Structured high-level intents	Proactive and reactive	Basic intent compilation, REST/CLI-based interfaces, ONOS based architecture	Limited expensiveness, lacks AI-driven optimization, reactive adaptation is low
IBN-Virtualization Platform Architecture [14]	High-level multi-tenant intents	Proactive & conflict-resolved	Multi-tenancy support, VN embedding, Intent-aware virtualization	Conflict resolution overhead, scalability concerns in large-scale deployments
IBN-automation for IOT Architecture (VISCR) [23]	Policy graphs, NLP-driven	Graph-based conflict detection	Vendor-independent intent translation, security policy automation	High computational cost for real time policy validation, dependency on vendor APIs
IBN-smart distribution grids architecture [21]	SLA & grid intent based policies	Closed-loop optimization & network slicing	5G network slicing, automated power grid orchestration	Latency concerns in real-time automation, lacks multi-domain policy coordination
IBN-Industrial Perspective framework [26]	High-level enterprise policies	AI/ML-based optimization & decision making	AI-driven intent execution, network-state observer, intent prioritization	Lacks of standardization for intent processing, scalability issues in large enterprises
Hey Lumi IBN	NLP-based natural	AI-driven entity	NER-based intent parsing, NILE	Ambiguities in natural language

architecture [16]	language intents	recognition & translation	language compilation, operator feedback refinement	interpretation, lacks robust policy enforcement and southbound programmability
IBN- connectivity & Cloud Architecture [28]	Service- based intent models	Intent mapping with AI/ML- driven service ranking	AI/ML-based intent refinement, cross-domain cloud network intent management	Interoperability issues across cloud providers, security vulnerabilities in cross-domain automation
OSDF – IBN framework [8], [9]	Application- aware policy- based intents	Hybrid: proactive rule installation + reactive adjustments	Advanced policy conflict resolution, Intra & Inter-site routing, Dynamic rule installation	High complexity in policy reasoning, lack of benchmarking for real-world scalability
P4 I/O IBN architecture [25]	Template- based intent specification	Runtime P4 program generation & modification	P4-based intent realization, DAG- based packet processing, Real- time pipeline updates	Requires expert knowledge of P4, limited support for AI-driven intent optimization

2.3 NLP-Driven Intent Processing in SDN

Intent-Based Networking (IBN) leverages Natural Language Processing (NLP) to translate high-level user intents into structured network policies, reducing the complexity of network management. Recent advancements explore NLP-driven frameworks to enhance intent processing, intent translation, validation, and assurance in SDN-based architectures.

One approach focuses on NLP-powered intent-based network management for private 5G networks, where user intents are matched to predefined workflows and configurations through a Functionality Template Catalog [20]. While this method

simplifies intent processing, it lacks dynamic workflow generation, limiting its flexibility in handling new and unforeseen user intents. Instead of autonomously creating new workflows, the system relies on predefined mappings for intent execution.

In contrast, LUMI employs a four-stage NLP-driven intent processing pipeline. The four stages are information extraction, intent assembly, intent confirmation, and intent deployment [16]. The information extraction module utilizes Named Entity Recognition (NER) techniques to extract entities. Those are a Bi-LSTM (Bidirectional Long Short-Term Memory) model for entity encoding and CRFs (Conditional Random Fields) for entity labelling. Extracted entities are then structured into Network Intent Language (Nile), ensuring syntactical correctness before intent deployment. Nile serves as an abstraction layer between natural language intents and network configuration commands. The intent confirmation stage iteratively refines the generated intent based on operator feedback, improving accuracy over time.

In another study, the intent conversion process make use of LSTM-based sequence-to-sequence models, where recognized entities are anonymized and transformed into numerical representations [24]. After that, Structured Network Intent Language (SNIL) refines network policies to ensure accurate translation and execution. SNIL is based on NILE, but modified and extended to distinguish between the intent for retrieving the network information and the intent for submitting the processed data to a destination network. Then, in the intent confirmation stage, operators verify and correct translated intents and feed corrections into the NLP model for continuous learning and adaptation.

An LLM-centric architecture for next-generation IBN management improves multi-domain intent decomposition and translation, simplifying network configuration across cloud, edge, and RAN domains [22]. The architecture uses LLMs to decompose high-level intents into domain-specific sub intents, which are then translated into Infrastructure-Level Intents (ILIs) using fine-tuned LLM modules. This methodology incorporates syntax, semantic, and correlation validation to ensure consistency. Furthermore, the system resolves intent conflicts, resource limitations, and activation constraints, making it adaptable across various infrastructure environments.

A structured approach to policy-based intent management was introduced in “Emergence”, a framework that progressively decomposes high-level intents into executable policies using LLMs with few-shot learning [11]. This three stage pipeline consists of intent classification, progressive decomposition, and validation. An iterative feedback loop ensures adaptive policy execution, while real-time monitoring via a MAPE-K (Monitor-Analyze-Plan-Execute-Knowledge) control loop dynamically adjusts policies in response to execution conditions. This system enhances intent realization by ensuring logical correctness before deployment. It utilizes another instance of an LLM trained with few-shot learning to verify whether the generated policies are correctly ordered and contain all necessary attributes.

Building on intent validation, a study on LLM-guided intent assurance proposes a framework to detect and mitigate intent drift—a scenario where a network’s operational state diverges from its intended configuration over time [12]. The system extracts Key Performance Indicators (KPIs) from intents and quantifies intent drift using an error function (E). The intent drift vector provides directional guidance for corrective actions, ensuring long-term compliance and network stability.

Table 2.2: Comparative Analysis of NLP-Based Intent Processing Approaches.

References	Intent Processing Stages	NLP / ML Techniques Used	Intent Representation	Limitations
[20]	Mapping user intents to predefined workflows	An NLP interface that converts natural intents into API calls	Functionality template (JSON, YAML)	Lacks flexibility in generating new workflows
[16]	Information extraction, intent assembly, confirmation,	NER (Bi-LSTM + CRF for identifying and labeling entities from	Network Intent Language (NILE)	Lacks ambiguity detection and intent conflict resolution.

	deployment	natural language intents)		
[24]	Classification, conversion, confirmation, implementation	LSTM-based sequence-to- sequence model	Structured network intent language (SNIL)	No intent optimizing mechanisms after deploying the intent
[22]	Decomposition, translation, negotiation, activation, assurance	LLM-based NLP processing	Infrastructure- Level Intents (ILIs)	Require fine- tuned LLMs
[11]	Classification, progressive decomposition, validation	Few-shot learning, feedback- driven policy generation	Structured policy tree	No intent drift detection
[12]	Intent drift detection, KPI extraction, assurance	Pre-trained LLMs, KPI- based error function	Formalized KPI-based intent	Requires historical KPI datasets

2.4 Intent Conflict Resolution

Intent conflict resolution is a critical challenge in Intent Based Networking (IBN), as multiple intents with competing objectives can lead to network inefficiencies and misconfigurations. Various approaches have been explored to address these conflicts, ranging from mathematical optimization techniques to policy refinement strategies and automated compiler-based solutions.

One approach to resolving conflicts in intent-driven networks leverages gradient-based optimization. Traditional methods struggle with dynamically obtaining loss

functions in complex environments, making it difficult to adjust conflicting network intents effectively. To address this, a novel gradient based framework was introduced that employs the Multiple Gradient Descent Algorithm (MGDA) to optimize multiple objectives simultaneously [6].

Beyond gradient-based methods, conflict resolution can also be approached through structured bargaining solutions. In the context of the Radio Access Network (RAN), direct conflicts arise when multiple intents require opposing adjustments to network parameters, such as antenna tilt. To address this, various bargaining solutions have been explored, extending beyond the traditional Nash Bargaining Solution (NBS) to include Weighted Nash Bargaining Solution (WNBS), Kalai Smorodinsky Bargaining Solution (KSBS), and Shannon Entropy Bargaining Solution (SEBS) [7].

Another approach focuses on refining intent translation to mitigate conflicts before they manifest in policy deployment. A proactive intent-refinement process has been proposed that incorporates machine learning and operator feedback to enhance accuracy and minimize inconsistencies in network policy implementation [17]. In this method, conflicts such as resource allocation mismatches (e.g., insufficient bandwidth) are detected during the translation phase, allowing operators to adjust configurations before deployment. The feedback loop further refines the process by learning from past conflicts, improving resolution accuracy over time, especially in cases where training data is limited. This adaptive mechanism ensures a more reliable intent-driven network management system.

Compiler-based approaches also play a vital role in intent conflict resolution, particularly when translating high-level network objectives into device-level configurations. Propane, a domain-specific language and compiler, simplifies the configuration of networks by allowing operators to specify routing policies using intuitive constraints [3]. To manage conflicting policies, Propane employs a hierarchical prioritization strategy where earlier-defined policy statements take precedence over later ones. Additionally, logical operators such as conjunction, disjunction, and negation are used to combine constraints while ensuring consistency. The compiler enforces regret-free preferences to maintain policy compliance under all failure scenarios and rejects ambiguous policies, requiring explicit resolution of

conflicts. This structured approach enhances the predictability and correctness of policy implementation in complex network environments.

Overall, intent conflict resolution in IBN is addressed through various methodologies, each offering distinct advantages. Gradient-based optimization ensures efficient multi objective adjustment, bargaining solutions introduce fairness considerations, machine learning-driven refinement enhances accuracy through adaptive learning, and compiler-based enforcement provides a structured mechanism for handling conflicts at the configuration level.

2.5 Machine Learning for SDN Optimization

The dynamic nature of modern networks makes it difficult for standard optimization techniques to adjust, therefore machine learning (ML) techniques are needed to improve SDN functionality. Recent developments in machine learning have shown great promise for optimizing SDN routing, network modelling, and traffic classification, leading to improved performance, adaptability, and scalability.

Traditional routing algorithms, such as shortest-path routing, exhibit slow convergence and congestion issues in dynamic network environments. Reinforcement learning (RL) has been considered as a potential solution to these issues, however, traditional RL techniques are limited by the computational cost and storage overhead of maintaining Q-tables. A Deep Reinforcement Learning (DRL) approach using the Deep Deterministic Policy Gradient (DDPG) algorithm has been proposed to enhance SDN routing [31]. The central idea is to leverage SDN's centralized control and network visibility to implement an intelligent routing mechanism that dynamically adjusts link weights based on network conditions. The DDPG framework includes an agent that modifies link weights to optimize data flow paths, an environment represented by the SDN network that provides feedback based on routing performance, a traffic matrix (TM) representing network load as the state, actions that adjust link weights to influence routing decisions, and a reward function that evaluates network performance metrics such as throughput and delay. The DDPG approach employs an actor-critic structure with experience replay and soft updates, which enhances training stability and convergence. This method significantly improves routing efficiency while reducing network maintenance costs.

Building upon the DDPG approach, an improved Deep Reinforcement Learning method, DDPG-EREP (Enhanced and Renewable Experience Pool), has been introduced to further optimize SDN routing [19]. This method dynamically adjusts the experience pool capacity and sample size based on iteration count, addressing the issue of outdated information in the experience pool. In DDPG-EREP, the state is represented by flow entries in the switch, actions involve modifying the queue order of flow tables via the northbound API, and the reward function is a throughput-based metric comparing byte transmission over time. Experimental results demonstrate that DDPG-EREP achieves faster convergence and better load balancing than standard DDPG approaches, enhancing overall network performance.

Table 2.3: Summary of DRL-Based Routing Approaches.

Approach	Key Technique	State Representation	Action	Reward Function
DDPG [31]	Deep Deterministic Policy Gradient	Traffic Matrix	Adjust the link weights	Throughput, delay
DDPG-EERP [19]	Enhanced experience replay	Flow entries in switch	Modify queue order	Byte transmission ratio

Network delay estimation plays a crucial role in network optimization. Traditional theoretical models, such as M/M/1 queuing models, often fail to capture the complexity of real world network conditions. To address this, machine learning based network modeling has been explored. A study compared the efficacy of an Artificial Neural Network (ANN)-based model with a theoretically inspired M/M/1-based model [5]. The ANN, implemented using the scikit-learn package with a sigmoid activation function, demonstrated superior accuracy in predicting network delays under varying traffic loads. The findings highlight the trade-off between interpretability and predictive accuracy in ML-based network modeling. The ANN-based approach proved more adaptable to real-world conditions than the queuing theory-inspired model.

Traffic classification is essential for network management and security. Traditional classification techniques, such as port based and Deep Packet Inspection (DPI), suffer from scalability and encryption-related challenges. Machine learning offers a promising alternative by leveraging flow-level data collection in SDN-enabled environments. A proposed architecture for collecting OpenFlow-based traffic data enables efficient traffic classification using supervised ML algorithms [2].

2.6 Intent Based Network Monitoring and Telemetry

Software-Defined Networking (SDN) has transformed network management by introducing centralized control, programmability, and automation [32]. However, effective SDN management depends on real-time telemetry, which provides deep visibility into traffic flows, device performance, and policy enforcement [15]. Unlike traditional static networks, making continuous monitoring essential. This enables detailed analysis, anomaly detection, and data-driven decisions for optimized performance, security, and congestion management. As SDNs expand, telemetry solutions must integrate seamlessly with both new and existing systems.

The traditional simple network management protocol (SNMP) has been the standard for monitoring and managing network devices in legacy hardware-driven architectures, offering a structured and widely adopted solution for monitoring network performance and fault detection [32]. It operates on a manager-agent model, where network devices run an SNMP agent that collects operational data and stores it in a Management Information Base (MIB), which can be accessed by a Network Management System (NMS) for configuration and monitoring. Although this approach has been effective in traditional networks, SNMP was designed for static, distributed environments and struggles to meet the demands of Software-Defined Networks (SDNs), which rely on centralized control, dynamic routing, and real-time telemetry for policy enforcement. The polling-based mechanism used in SNMP introduces latency and additional network overhead, making it inefficient to capture the highly dynamic state of SDN traffic flows and OpenFlow events, which require instantaneous adaptation to network conditions [15].

To address these limitations, the SDN Management Protocol (SDNMP) extends SNMP by integrating it with SDN controllers, allowing traditional NMS platforms to

access SDN telemetry data through a unified SNMP interface. Unlike conventional approaches that require a complete overhaul of network management infrastructure, SDNMP transparently stores OpenFlow events into MIBs, enabling SNMP-based tools to retrieve SDN data as if they were managing a legacy network. This hybrid approach bridges the gap between legacy NMS systems and modern SDN architectures, ensuring compatibility, reducing operational complexity, and facilitating SDN adoption while enhancing visibility into both physical and virtual SDN networks. By maintaining SNMP as the interface for network monitoring, SDNMP allows network operators to transition to SDN-based management with minimal disruption, ensuring continued use of established monitoring frameworks while gaining access to SDN's programmable capabilities.

A more advanced approach to real-time SDN telemetry is In-Band Network Telemetry (INT), which embeds monitoring metadata directly into user traffic as it traverses the network [13], [15], [18]. Unlike traditional telemetry techniques that rely on periodic polling or active probes, INT ensures per packet and per-hop visibility by appending monitoring data at each transit node. This technique eliminates artificial probe packets, reducing additional monitoring overhead, and captures real-time network state at the precise moment traffic flows through a device. Additionally, it provides fine-grained control over the amount and type of monitoring data collected, making it a more efficient solution for SDN environments. The collected telemetry information is extracted at egress points and forwarded to a monitoring host for analysis, ensuring end-to-end network visibility without injecting additional traffic. Kim et al. demonstrated the capabilities of INT using the P4 programming language, demonstrating its effectiveness in identifying performance bottlenecks, diagnosing latency spikes, and optimizing network configurations. Unlike legacy monitoring systems, INT provides real time insights into network performance, making it a crucial technology for modern SDN-based infrastructures.

Machine learning and artificial intelligence (AI) further enhance intent-based telemetry by enabling automated anomaly detection and traffic optimization. AI-driven techniques analyze large volumes of telemetry data to detect suspicious activities, security threats, and network inefficiencies in real time [4], [30]. Deep Neural Networks (DNNs) have demonstrated significant improvements in anomaly detection accuracy, reducing false positives compared to traditional signature-based

detection methods. These advancements contribute to proactive network security and self-optimizing SDN infrastructures, making AI an essential component of future telemetry solutions.

Despite these advancements, several challenges remain in intent-based telemetry. Embedding telemetry data in real traffic increases packet size and processing overhead, potentially leading to network congestion and fragmentation issues [10], [13], [15]. Additionally, security concerns arise, as telemetry metadata could be exploited for traffic analysis attacks, posing privacy and confidentiality risks. Ensuring secure telemetry data transmission through encryption and access control mechanisms is essential to mitigating these threats. Interoperability also remains a critical challenge, as different SDN controllers and devices use varying telemetry standards. Establishing standardized APIs and cross-platform compatibility is necessary to ensure seamless telemetry integration across heterogeneous network environments.

Future advancements in intent-based telemetry will focus on enhancing AI-driven automation, improving security, and optimizing scalability. AI-powered telemetry analytics will further refine network monitoring by enabling predictive anomaly detection and intelligent traffic routing. Automated intent translation mechanisms, leveraging Natural Language Processing (NLP), will simplify the process of converting high-level network objectives into actionable policies. Additionally, strengthening encryption and authentication protocols will secure telemetry data against potential cyber threats, ensuring that network monitoring does not introduce new vulnerabilities. As networks continue to evolve, intent based telemetry will play a vital role in creating intelligent, responsive, and self-adaptive infrastructures.

2.7 Summary

In this chapter, we examined the role of Natural Language Processing (NLP) and Machine Learning (ML) in intent-based networking (IBN) for Software-Defined Networks (SDNs), focusing on their contributions to intent translation, conflict resolution, and network optimization. Our analysis highlights that while IBN architectures are evolving towards more dynamic and adaptable network ecosystems, challenges such as the gap between natural language intent expression and SDN rule

execution, the lack of standardized frameworks, and the complexity of real-time intent verification remain significant obstacles. Additionally, intent conflict resolution mechanisms are still under development, requiring further refinement to ensure reliability in diverse network environments. Despite these challenges, advancements in artificial intelligence (AI), particularly the integration of large language models (LLMs), present promising opportunities to enhance intent processing, automation, and overall network efficiency. Addressing these gaps through collaborative research and standardized methodologies will be crucial in realizing the full potential of NLP-driven IBN in SDN management.

Chapter 3 : Reshaping SDN: Harnessing NLP and ML to Empower Intent-Based Networking

3.1 Introduction

Recent advances in Intent-Based Networking (IBN) have primarily been driven by Natural Language Processing (NLP) breakthroughs. As discussed in Chapter II, NLP, particularly through the use of Large Language Models (LLMs), has shown promising results in IBN. The open-source availability of pre-trained LLMs, combined with techniques like few-shot learning allows for the automated generation of SDN configuration policies. This chapter presents the key technologies that can be adapted to develop an NLP-driven intent-based networking layered architecture for SDNs. The research integrates Natural Language Processing (NLP), Machine Learning (ML), and Software-Defined Networking (SDN) to create an automated and intelligent networking framework. NLP enables the system to interpret human-expressed intents and translate them into network policies. ML-based intent optimization ensures that network policies are dynamically refined based on past experiences and real-time network conditions. SDN provides a programmable infrastructure that enforces and executes network configurations based on user-defined intents.

3.2 Natural Language Processing (NLP) for Intent Understanding, Named Entity Recognition and Policy Generation

NLP plays a fundamental role in converting high-level user intents into network policies that an SDN controller can understand. Traditional networking relies on complex, low-level rule configurations, which makes manual intervention inconvenient. With NLP, users can specify network requirements in natural language (e.g., *"Ensure low latency for video traffic"*) instead of manually defining QoS parameters.

Earlier adaptations of IBN frameworks relied on predefined templates to capture user input and convert it into network policies. This approach provided users with a limited ability to express high-level intents within the IBN framework. However, these frameworks lacked the flexibility to generate network policies for arbitrary scenarios. With advancements in NLP, the first IBN frameworks leveraging LLMs were primarily used to identify entities within user intents and map them to predefined workflows [16]. Now, with the fine-tuning of LLMs,

NLP can be used to generate entirely new network configuration policies, enabling greater adaptability and automation in intent-based networking [8], [9].

The recent success of Large Language Models (LLMs), such as GPT-4, DeepSeek, and Llama, has significantly improved the ability of machines to understand human language. LLMs enable intent-based networking by,

- Parsing and understanding network intents expressed in natural language
- Mapping high-level intent expressions to predefined network functions
- Disambiguating conflicting intents using contextual understanding

Pre-trained LLMs combined with few-shot learning techniques enable rapid adaptation to new networking scenarios without extensive retraining. Few-shot learning allows the model to generalize network intents using minimal labelled data [11], [22], [24].

Named Entity Recognition (NER) is a subtask of Natural Language Processing (NLP) that involves identifying and classifying key entities in a text. Entities can belong to predefined categories such as names of people, organizations, locations, dates, numerical values, and domain-specific terms. In the context of Intent-Based Networking (IBN), NER is crucial for extracting network-related entities from user-expressed intents. For example, given the user intent: "Ensure high-priority traffic for video conferencing between New York and London with a minimum bandwidth of 100 Mbps and latency below 50 ms," an NER model would extract key networking parameters such as the application type (video conferencing), locations (New York, London), bandwidth requirement (100 Mbps), and latency constraint (50 ms). By structuring this extracted data, the system can automatically map user intents to appropriate SDN configurations, enabling seamless network policy enforcement without manual intervention.

3.3 Machine Learning for Intent Optimization

While NLP enables intent extraction, machine learning ensures that generated network policies are optimal and dynamically adaptable. In real-world deployments, network conditions constantly change, requiring continuous policy optimization.

Reinforcement Learning (RL) can be used for Dynamic Network Optimization. RL plays a key role in adapting SDN configurations based on real-time performance feedback. RL agents learn to adjust routing, traffic prioritization, and load-balancing strategies without human intervention. The system can observe network metrics (e.g., latency, jitter, bandwidth utilization) and adjust SDN configurations dynamically to meet intent objectives [19], [31].

Other than Reinforcement Learning for dynamic network optimization Supervised Learning models can be for intent prediction. Supervised learning models can be trained on historical intent data to predict the best network configurations for new intents. Using datasets of past network requests and performance logs, models can anticipate user requirements and automate SDN decision-making. Using datasets of past network requests and performance logs, models can anticipate user requirements and automate SDN decision-making.

3.4 Evaluating NLP-Driven Intent-Based Networking Through SDN Emulation

To assess how well natural language intents can be translated into SDN policies, the efficiency of automated network orchestration and the overall performance impact of an NLP-driven approach SDN emulation tools are required [17], [24].

To evaluate the proposed NLP-driven IBN architecture, a controlled SDN emulation environment will be set up using open-source SDN emulation tools. The key tools to be utilized include,

- Mininet: A widely used network emulator that simulates an SDN environment with virtual switches and hosts.
- Ryu / ONOS / OpenDaylight: SDN controllers that will process the NLP-derived network policies and enforce them on the emulated network.
- Open vSwitch (OVS): A software switch that enables programmable SDN flow control.
- P4 Runtime: For advanced packet processing, allowing fine-grained SDN policy enforcement.

The emulated network topologies will include multiple hosts, switches, and SDN controllers, ensuring realistic conditions for evaluating intent-driven automation.

3.5 Summary

This chapter outlined the core technologies that can be adopted for this research. NLP enables intent extraction and policy generation while ML optimizes policy generation, and SDN emulation environments can be used to enforce configurations dynamically. However, as the research progresses, these identified technologies may evolve, adapt, or be refined based on experimental findings, emerging advancements, and the specific requirements of the implemented system.

Chapter 4 : An NLP-Driven Approach to Intent Interpretation for SDN Configuration

4.1 Introduction

Intent-based networking is transforming how network configurations are managed by enabling users to define policies in natural language, which are then translated into structured, executable SDN commands. This research focuses on developing an Intent Processing Framework that leverages Large Language Models (LLMs), and Machine Learning (ML) techniques to accurately interpret user intents and generate network configurations dynamically. By integrating adaptive learning and real-time telemetry feedback, the proposed system ensures intelligent automation, policy conflict resolution, and optimized resource allocation, making SDN management more efficient and scalable.

4.2 Intent Processing Flow

The Intent Processing Framework is a crucial component in transforming user-defined natural language intents into structured, executable configurations within an SDN environment. The following sections describe the various stages involved in processing and executing user intents efficiently.

The process begins with User Intent, where users express their network requirements in natural language, such as “Prioritize VOIP during work hours” or “Block access to specific IPs.” Since human language is inherently ambiguous, these intents must be carefully interpreted and structured. This is achieved through Natural Language Processing (NLP) techniques like Named Entity Recognition (NER), which work together to extract key components from the user’s input, such as network objects (IP addresses, user groups, devices), actions (prioritize, block, allow), time constraints (work hours, specific dates), and QoS parameters (bandwidth, priority levels). NLP helps in understanding sentence structures, disambiguating meanings, and ensuring contextual accuracy, while NER focuses on identifying and categorizing specific

named entities. This step ensures that all relevant intent components are extracted accurately.

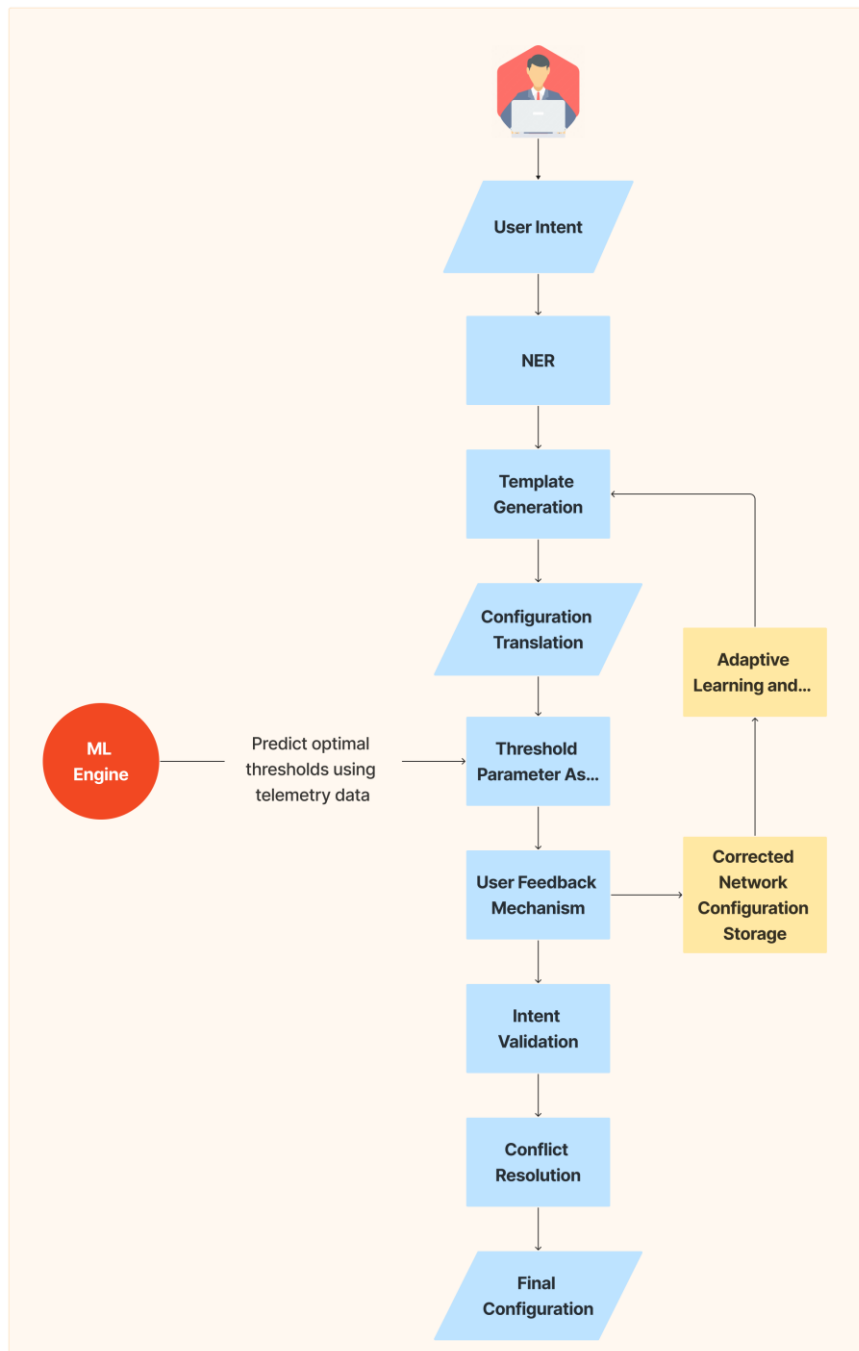


Figure 4.1: Intent Processing Flow.

Once the intent components are identified, the Template Generation phase dynamically generates network policy templates using Natural Language Processing (NLP) techniques. NLP helps in structuring the extracted entities into a format suitable for network configuration.

Following template generation, the Configuration Translation phase converts these structured templates into machine-readable network commands. The extracted policies are transformed into JSON format or direct SDN commands that can be interpreted by controllers like ONOS. This enables seamless integration with SDN switches, ensuring that the intended network policies are applied correctly.

The next step involves Threshold Parameter Assignment, where key parameters such as minimum/maximum bit rates, priority levels, and bandwidth allocations are determined using a machine learning (ML) models trained on network telemetry data. This ML-driven approach enables dynamic threshold prediction based on real-time network conditions, optimizing resource allocation and ensuring efficient performance. Similarly, ACL rules are refined with priority levels that adapt to evolving network traffic patterns, enhancing security and performance.

User involvement is critical in maintaining accuracy, which is why the User Feedback Mechanism allows users to review applied configurations and provide input on their effectiveness. If configurations do not fully meet user expectations, feedback is collected to refine the processing pipeline, ensuring continuous improvement and adaptability.

To enhance performance over time, the framework includes an Adaptive Learning System. This component gathers user input on configurations and stores it, enabling the LLM to continuously learn and refine intent processing. This ensures that future intents are generated with greater precision and accuracy.

Before final deployment, the Intent Validation and Compliance phase verifies whether the interpreted intents align with syntax requirements and network policies. This validation step prevents errors before implementation, ensuring that generated configurations meet user expectations and adhere to existing network guidelines.

One of the critical challenges in intent-driven networking is handling conflicting policies. The Policy Conflict Resolution phase detects and resolves inconsistencies between different intents. The system determines the optimal policy based on predefined rules, optimization techniques, or user intervention. This phase ensures that all policies are implemented in a conflict-free manner.

Finally, in the Automated Deployment and Execution stage, the validated policies are deployed to the network. The SDN controller, using REST or gRPC-based APIs, pushes these configurations to SDN-enabled switches, ensuring seamless application of ACL rules, QoS settings, and routing policies. This automation eliminates manual intervention and enhances network agility and efficiency.

By implementing this structured framework, our system enables a robust intent-driven SDN architecture, where network configurations are dynamically generated, validated, and optimized based on user-defined intents. This approach ensures high accuracy, adaptability, and automation, significantly improving modern network management.

4.3 SDN Execution Flow

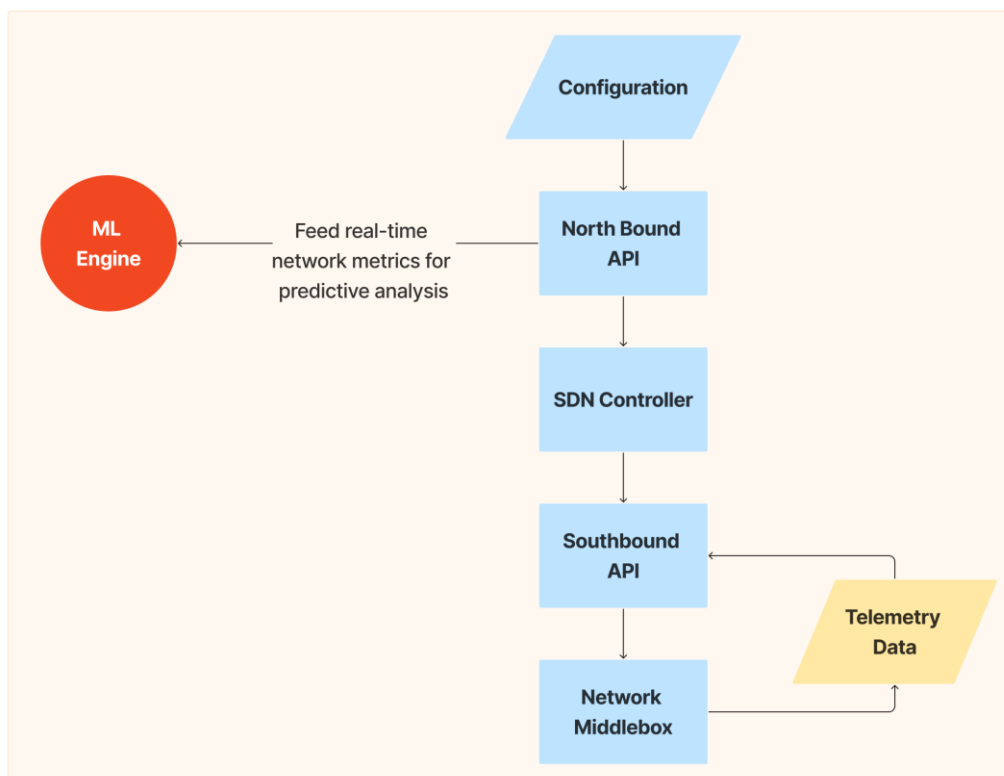


Figure 4.2: SDN Execution Flow

The SDN Execution Pipeline operates as a seamless and dynamic flow of processes that work together to enable intelligent, responsive, and self-optimizing network management. This pipeline begins with the generated network configurations from the

intent process flow. SDN controller's North Bound API allows higher-level applications, such as network management tools to send configurations or intents to the SDN controller.

Once the configurations are received by the SDN controller, they are converted into low-level commands that network devices, such as switches, routers, and firewalls, can understand and implement. These commands are transmitted to the network devices via the South Bound Interface (SBI). The SBI is the communication channel between the SDN controller and the network infrastructure, enabling the controller to send configuration commands, as well as collect telemetry data from network devices.

As network devices implement the configurations, they continuously generate telemetry data, which includes real-time performance metrics such as traffic flow, bandwidth utilization, latency, packet loss, and other relevant network statistics. This telemetry data is then sent back to the SDN controller, where it is analyzed. The telemetry data provides valuable insights into the current state of the network, identifying areas where performance may be degraded or where adjustments are needed to meet user-defined intents.

The role of telemetry data is crucial, as it feeds directly into the Intent Generation Process to refine and optimize network configurations. By analyzing this real-time performance data, the SDN controller can make informed decisions about the network's behavior. Machine learning models or predictive algorithms can be applied to this telemetry data to dynamically adjust and predict threshold values for network performance, such as setting bandwidth limits or defining latency tolerances. These predicted thresholds are used to adjust the configuration, ensuring that network performance remains within the desired parameters.

This creates a continuous feedback loop where telemetry data not only informs real-time network optimization but also drives the prediction of future network thresholds. As the SDN controller monitors the evolving state of the network, it can adapt the configurations and predict when certain performance parameters will exceed predefined thresholds. For example, if latency increases due to traffic congestion, the SDN controller can predict when the congestion will reach a critical threshold and proactively adjust the QoS settings to prioritize essential traffic.

Through this integrated process, the SDN Execution flow provides a high level of automation and adaptability. The flow allows the SDN system to operate with minimal manual configuration, as the network adjusts its settings in real time based on the evolving needs of the system and user-defined goals. As new data is collected, the system can predict future performance requirements, adjust configurations, and ensure that the network operates efficiently, and within optimal performance ranges at all times.

4.4 Evaluation

In this study, we aim to compare the performance of a traditional large language model (LLM)-generated network configurations with a custom fine-tuned LLM-generated network configurations tailored specifically for Software-Defined Networking (SDN) configurations generation. The primary goal of this evaluation is to determine which approach is better suited for accurately interpreting user intents and generating network configurations that align with SDN requirements. The comparison will be based on several key aspects, including accuracy, efficiency, relevance, scalability, and user satisfaction.

The accuracy of each system will be assessed by evaluating how well they recognize and extract key entities from user intents, such as IP addresses, time ranges, and QoS classes. Additionally, the correctness of the configurations generated will be measured against a manually annotated ground truth dataset. By comparing the percentage of correctly generated configurations, we can determine which system is more reliable in producing accurate network configurations.

Efficiency is another crucial factor in this evaluation. The response time of each system will be measured to understand how quickly they generate configurations based on given intents.

Relevance plays a significant role in ensuring that the generated configurations are useful and practical for SDN environments. The domain-specific relevance of each system's outputs will be analyzed to determine whether they meet the specific requirements of SDN tasks, such as Quality of Service (QoS) management and Access Control List (ACL) policies. Furthermore, the error rate will be measured to

assess the frequency of irrelevant or incorrect configurations generated by each system. A lower error rate indicates that the system produces more reliable outputs.

Scalability is another key consideration in this evaluation. Both systems will be tested under increasing workload conditions to determine how well they handle a high volume of user intents. The performance of each system under heavy loads will be measured in terms of response time and accuracy to ensure that they remain effective and efficient as the complexity of the input increases. A highly scalable system will be able to maintain its performance and accuracy even when processing large numbers of requests.

Additionally, we will evaluate how well the models translate user intents into accurate network configurations by analyzing fluctuations in the throughput of network switches. To simulate real-world conditions, we will introduce noise into the system before the user provides intent. If the system can effectively mitigate the noise, the throughput should significantly decrease, and the network should recover once the translated configurations are deployed. This will serve as an indicator of how well the system interprets and applies user intents.

Through this evaluation, we aim to provide a data-driven assessment of the feasibility and effectiveness of each approach. The results of this study will offer valuable insights into how AI-driven NLP models can be leveraged for SDN configuration generation and whether a specialized NLP approach is necessary to achieve optimal results.

4.5 Summary

This study presents a structured approach to intent-driven SDN configuration, detailing each stage from user intent recognition to automated deployment. The proposed framework combines NLP techniques to generate dynamic network configurations, ML-based threshold assignment, and a continuous feedback loop for adaptive improvements. Additionally, an evaluation is conducted to compare the effectiveness of a general-purpose LLM versus a custom NLP pipeline for SDN configuration generation, measuring accuracy, efficiency, relevance, and scalability. The results aim to determine the most suitable approach for achieving precise, automated, and scalable intent-based networking solutions.

Chapter 5 : Analysis and Design

5.1 Introduction

The proposed system integrates AI-driven intent recognition with SDN (Software-Defined Networking) to create an intelligent and adaptive networking framework. The architecture is designed to interpret natural language user intents, process them through multiple intelligent layers, and enforce them dynamically using SDN controllers and programmable middleboxes. The previous chapter thoroughly discussed **the NLP-Driven Approach to Intent Interpretation for SDN Configuration**, outlining how natural language processing enables seamless user interaction with the network. Building upon that foundation, this chapter delves deeper into **the novel proposed IBN (Intent-Based Networking) Architecture**, detailing its key components, data flow, and the mechanisms that enable dynamic and intelligent network intent management.

5.2 System Overview Diagram

To provide a comprehensive understanding of the proposed **Intent-Based Networking (IBN) Architecture**, the following **High-Level Design Diagram** illustrates the key components and their interactions. This architecture is structured into multiple layers, each responsible for processing and enforcing user-defined intents within an SDN-driven environment. The diagram visually represents the data flow from **natural language input to policy enforcement and feedback loops**, ensuring seamless adaptability and optimization. Below, we present the system's high-level design, followed by a detailed breakdown of its functional layers.

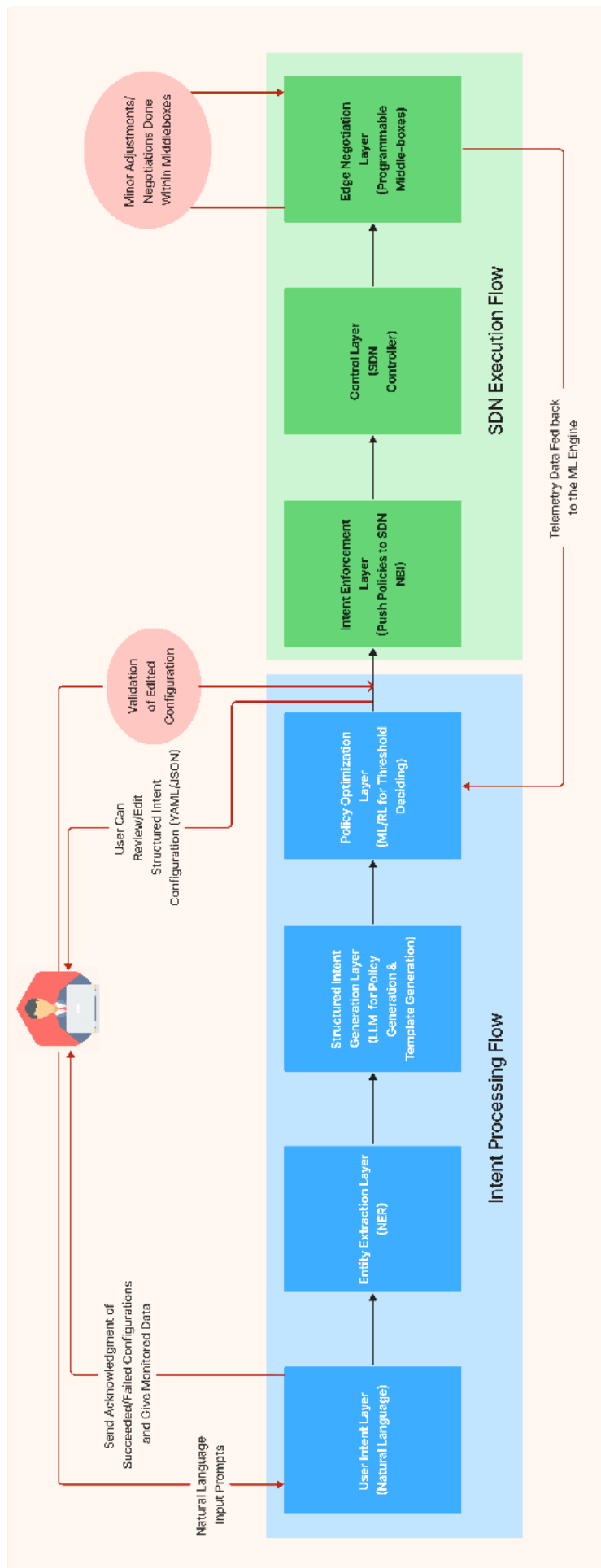


Figure 5.1: High Level Design of the Proposed NLP-Driven IBN Layered Architecture

5.3 Module Breakdown

The proposed Intent-Based Networking (IBN) Architecture consists of multiple interconnected modules that work together to process user-defined intents, optimize policies, and enforce network configurations dynamically. Each module plays a crucial role in ensuring a seamless and intelligent networking experience. Figure 5.1 illustrates the high-level design of the system, showing how data flows through different layers to achieve adaptive SDN configuration. Below is a detailed breakdown of these key modules.

1. User Intent Layer

This serves as the entry point for network administrators to define high-level policies in natural language. It preprocesses the input and forwards it to the NLP layers for further analysis. This module ensures that user-defined intents are captured accurately and efficiently, setting the foundation for subsequent processing.

2. Entity Extraction Layer

This Layer is responsible for extracting relevant entities from user inputs. Using AI-driven entity extraction techniques, it identifies whether the user prompt is a configuration deployment or a query regarding network details. Moreover, it extracts key parameters such as latency, bandwidth, and traffic types. The extracted information is then converted into structured data and passed to the structured intent generation module for further refinement.

3. Structured Intent Generation Layer

This module leverages Large Language Models (LLMs) to map extracted entities to predefined network policy templates. This module generates structured policies in formats such as YAML or JSON, allowing users to review and validate them before enforcement. By ensuring that natural language inputs are correctly translated into enforceable policies, this module bridges the gap between user-defined intents and network configuration.

4. Policy Optimization Layer

The ML/RL-Based Policy Optimization Layer enhances decision-making by optimizing policy parameters through Machine Learning (ML) and Reinforcement Learning (RL). This module ensures that thresholds and constraints are resolved dynamically while considering historical data and real-time network conditions. By continuously learning from telemetry data, it refines policies to improve network efficiency and adaptability.

5. Intent Enforcement Layer

The Intent Enforcement Layer is responsible for translating optimized policies into Software-Defined Networking (SDN)-compatible rules which are flow rules, routing configurations, QoS policies and ACL rules. Once the structured policies are finalized, this module pushes them to the Northbound Interface (NBI) of the SDN controller, ensuring that intent-based policies are dynamically enforced across the network.

6. Control Layer

The Control Layer plays a pivotal role in executing the intent-driven policies. This is where the SDN controller resides. The SDN controller dynamically executes received SDN rules and continuously monitors the network state. It collects through the SBI (South Bound Interface) real-time performance metrics, which are sent back to ML and NLP governing layers as telemetry data to improve decision-making in future policy optimizations.

7. Edge Negotiation Layer

This module fine-tunes network configurations within programmable middleboxes. It ensures that policies are adjusted at the edge to maintain adherence to global network rules. Additionally, it resolves conflicts before final enforcement and feeds back telemetry data to the ML/RL-Based Policy Optimization Module for continuous learning and refinement.

Moreover, the proposed architecture, as shown in Figure 5.1, incorporates multiple feedback loops to ensure continuous refinement of network policies. After structured intents are generated, users have the opportunity to review and edit configurations before enforcement, ensuring alignment with network requirements. Structured intent validation further verifies that policies adhere to predefined templates and constraints. Telemetry feedback from SDN controllers and middleboxes enables dynamic policy adjustments, allowing for minor refinements and negotiations within programmable middleboxes to optimize enforcement at the edge. Additionally, the system provides users with explanations of the generated configurations, offering transparency into the decision-making process and allowing for informed modifications if needed.

Together, these 7 layers create an intelligent and adaptive SDN-based networking framework, enabling a seamless translation of user-defined intents into enforceable, optimized network policies.

5.4 Summary

This chapter outlined the proposed Intent-Based Networking (IBN) Architecture, integrating Natural Language Processing and Machine Learning into Software Defined Networks for intelligent and adaptive network management. The system processes natural language intents, optimizes policies, and enforces them via SDN controllers and middleboxes. A high-level diagram (Figure 5.1) illustrated the seven key layers, from intent extraction to policy enforcement. Feedback loops enable user validation, telemetry-driven refinements, and edge negotiations, ensuring adaptability and transparency. This approach enhances network intelligence, scalability, and efficiency, aligning configurations with real-time conditions.

Chapter 6 : Implementation

6.1 Introduction

This chapter outlines the proposed implementation of the Intent-Based Networking (IBN) system, focusing on network intent interpretation, policy enforcement, and adaptive learning mechanisms. The implementation plan is based on the high-level implementation design depicted in Figure 6-1. Since this research is ongoing, technologies and methodologies may evolve based on practical evaluations and experimental findings.

6.2 Emulation Setup Description and Topologies

To validate the proposed system, a network emulation environment will be built using SDN-controlled topologies with automated traffic generation. This section breaks down the technologies to be used and the proposed implementation plan.

6.2.1 Proposed Key Technologies and Justifications

To achieve the goal of an intelligent and adaptable IBN architecture, the following technologies are considered:

- NLP Model (DeepSeek, GPT, LLaMA) → For Natural Language Understanding (NLU)
 - DeepSeek is the most likely choice due to its reasoning capabilities and open-source flexibility.
 - GPT & LLaMA are considered as alternatives for evaluating different reasoning abilities and few-shot learning performance.
- Few-Shot Learning for Intent Extraction
 - Ensures the system can interpret complex user intents with minimal training data, reducing reliance on large, pre-labeled datasets.

- Reinforcement Learning (RL) for Policy Tuning
 - Used to optimize policy selection and adaptation based on feedback from the emulated network environment.
 - Helps refine decision-making by continuously learning from SDN telemetry and policy enforcement results.

- Mininet/Containernet/P4 for Emulation
 - Mininet and Containernet provide scalable virtualized network environments for testing policy deployment. Containernet is used because it can emulate more realistic virtual machines that run real operating systems and services, unlike Mininet hosts, which are limited to lightweight process-based emulation.
 - We propose using P4 because it enables Inband Network Telemetry (INT) monitoring, allowing real-time visibility into network performance and traffic flow analysis. Moreover, it would enhance the programmability in middleboxes which would help enhancing the proposed IBN framework.

- SDN Controllers (ONOS, OpenDaylight) → For Policy Enforcement
 - ONOS is currently the preferred choice due to comprehensive NBI (Northbound Interface) documentation and robust API support.
 - OpenDaylight remains an alternative for testing controller performance variations.

6.2.2 High-Level Implementation Plan

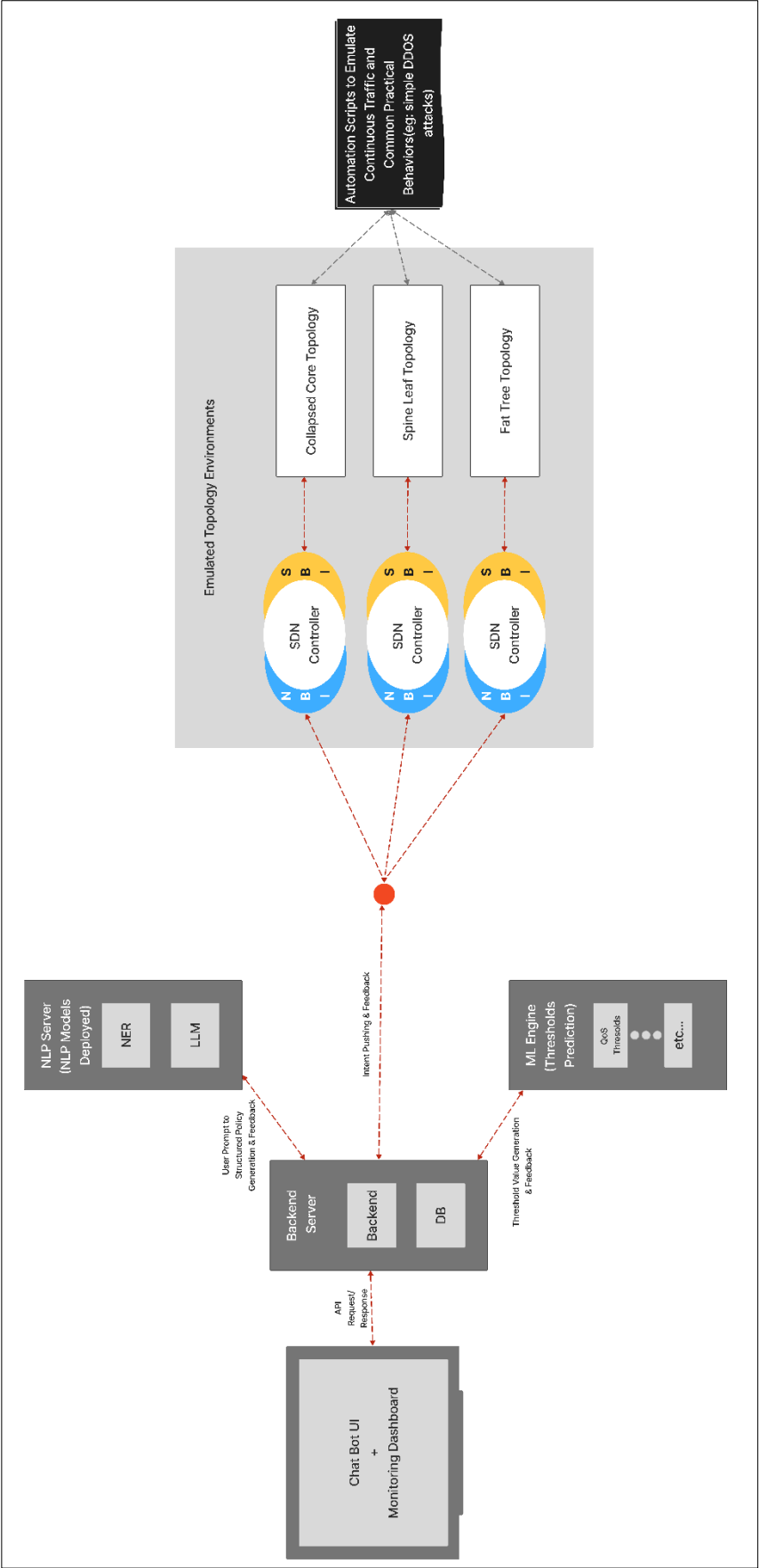


Figure 6.1: High Level Implementation Plan

1. User Intent Processing & Policy Generation
 - Users interact with a Chatbot UI + Monitoring Dashboard.
 - The NLP Server (DeepSeek/GPT/LLaMA) converts natural language prompts into structured network policies.
 - These policies are validated and stored in the Backend Server + DB.
2. Intent Execution in Emulated SDN Topologies
 - The SDN Controllers (ONOS/OpenDaylight) enforce the generated policies.
 - Network behaviors are tested on three key topologies:
 - Collapsed Core: This topology is typically used in smaller organizations with fewer buildings, offering a simplified, cost-effective solution with limited scalability for smaller network needs.
 - Spine-Leaf: This topology is typically used in large entities with multiple buildings, providing high availability, fault tolerance, and scalability for complex, distributed network structures
 - Fat Tree: This topology is typically used in high-performance computing environments like universities or research institutions, offering scalable, high-bandwidth, and low-latency connectivity for large data workloads.
3. Real-Time Policy Adjustment Using ML & RL
 - A Machine Learning Engine predicts QoS thresholds based on network state and telemetry.
 - Reinforcement Learning (RL) fine-tunes policy adjustments dynamically.
4. Traffic Generation & Attack Simulation
 - Automation scripts will generate realistic network traffic.
 - Practical scenarios like DDoS attacks, congestion, and QoS degradation will be tested.

These proposed implementations, including the three topologies and NLP & ML pipelines, will be hosted on the Computer Engineering Department's resources for optimal support and infrastructure.

6.2.3 Proposed Topologies

The following are the proposed three topologies to be implemented regarding this emulated setup. The design of these topologies is done in such a way that they encompass all the basic network methods such as routing, switching, NAT, VLANs, and security measures. Additionally, these topologies will be used to evaluate the performance and scalability of the Novel IBN framework, ensuring they can support a variety of network configurations and security threats in realistic environments. Each topology is chosen to address specific use cases, such as high-performance computing, enterprise scalability, and cost-effective network solutions, providing a comprehensive testing ground for the framework's capabilities.

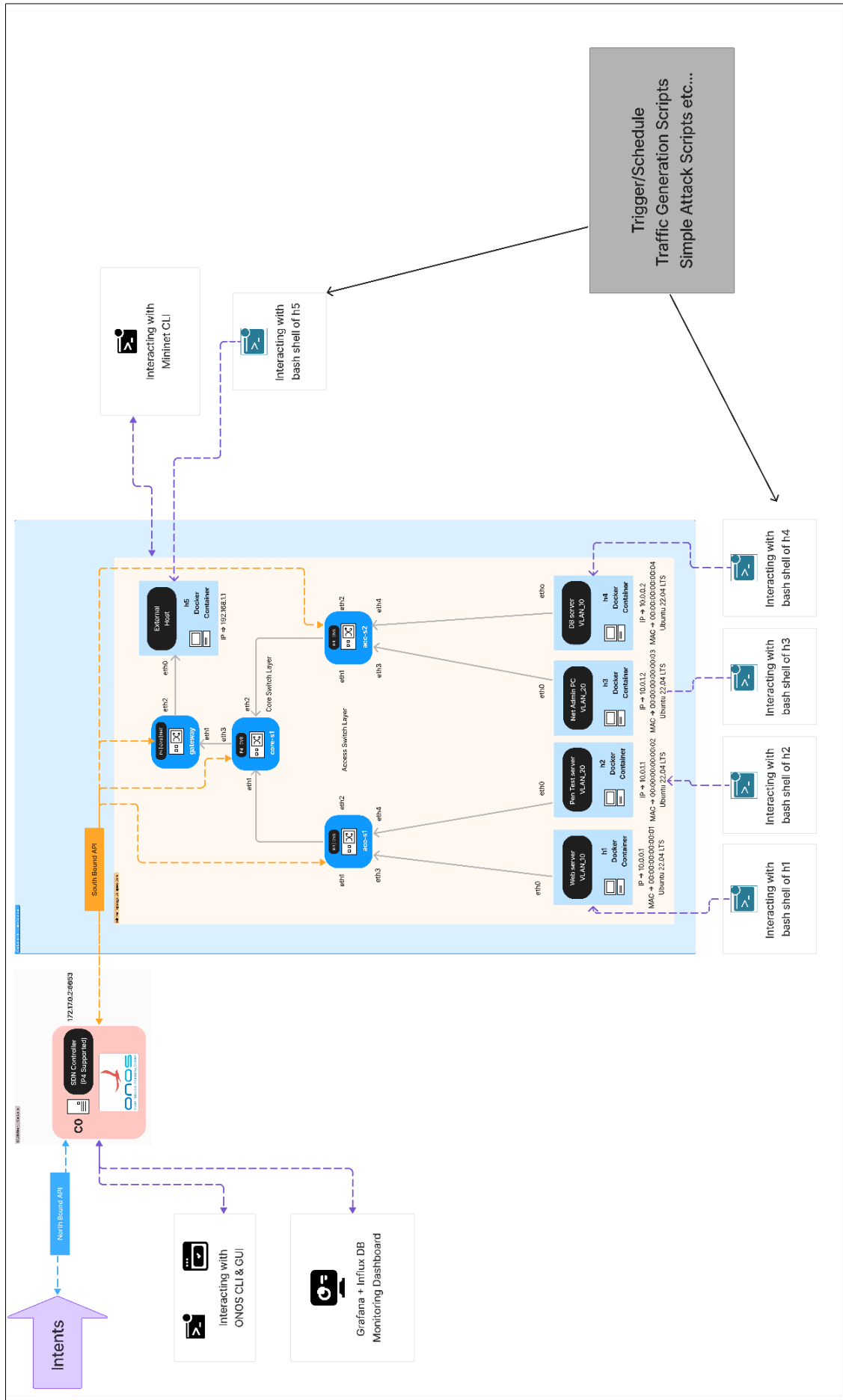


Figure 6.2: Proposed Collapsed Core Topology

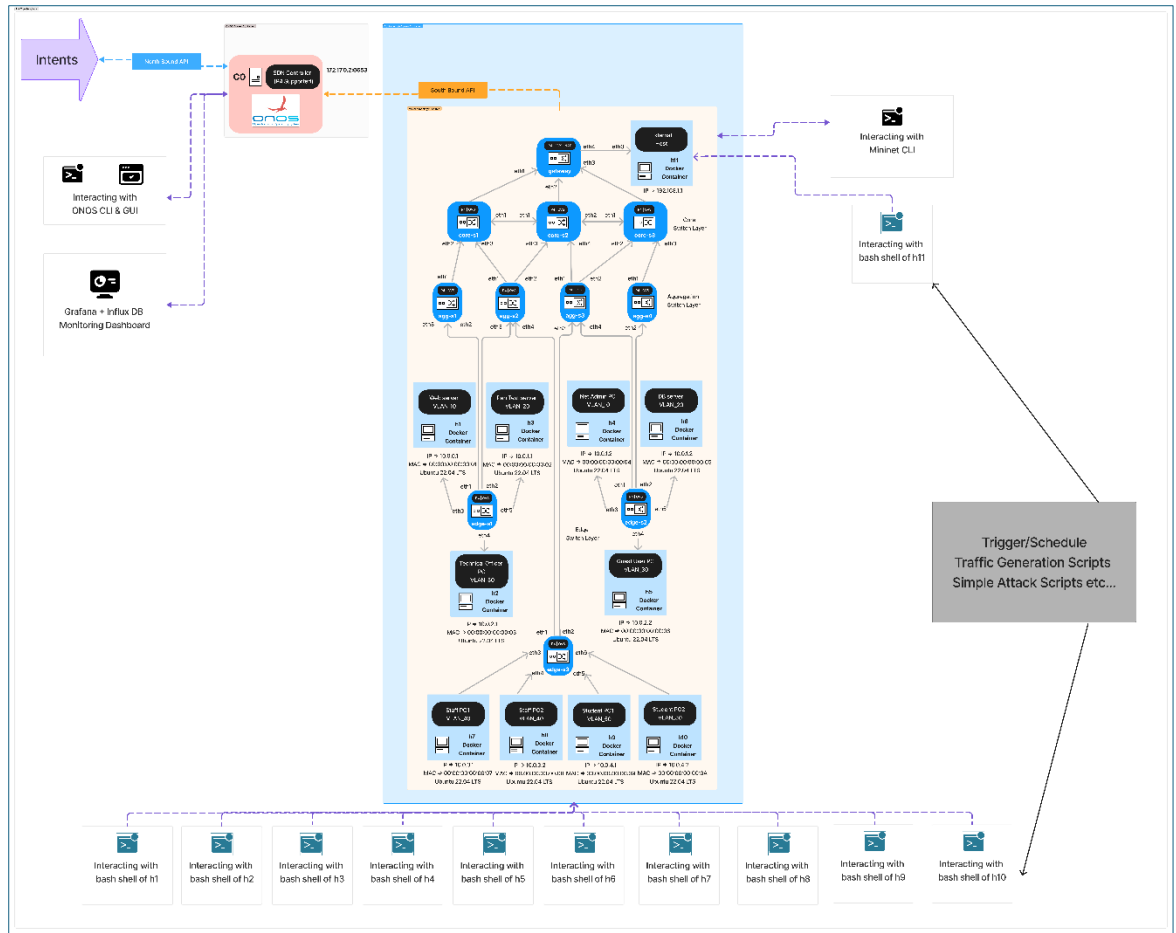


Figure 6.4: Proposed Fat-Tree Topology

6.3 Summary

This chapter outlines the implementation plan for the proposed Intent-Based Networking (IBN) architecture, detailing the technologies, emulation setup, and the three key network topologies chosen for testing. The proposed IBN architecture integrates Natural Language Processing (NLP) models, few-shot learning for intent extraction, reinforcement learning for policy tuning, and SDN-controlled emulation using Mininet, Containernet, and P4. The SDN controllers (ONOS, OpenDaylight) will enforce generated policies, while real-time policy adjustments will be made using Machine Learning (ML) and Reinforcement Learning (RL) to dynamically refine network operations. The chapter highlights three network topologies—Collapsed Core, Spine-Leaf, and Fat Tree—each selected to validate the scalability and performance of the IBN framework across different network use cases. Automated traffic generation and attack simulations will further test the system's robustness.

These implementations will be hosted on the Computer Engineering Department's resources, ensuring the necessary infrastructure for comprehensive testing and evaluation.

Chapter 7 : Discussion

7.1 Introduction

This chapter provides a critical assessment of the proposed NLP-driven intent-based networking framework within the broader context of Software-Defined Networking (SDN) and Intent-Based Networking (IBN). By analyzing how our approach compares to existing frameworks, we highlight its advantages, limitations, and areas where it improves upon prior methodologies. Additionally, we explore key research directions that can further refine and expand the capabilities of NLP-driven intent processing in SDNs.

7.2 Comparison with Existing Work

Most traditional IBN architectures, such as ONOS IBN Framework [1], [33], rely on structured high-level intents processed using proactive and reactive mechanisms. While these frameworks provide basic intent compilation, they suffer from limited expressiveness and slow reactive adaptation. Our framework advances natural language intent processing using NLP-driven models similar to Hey Lumi IBN Architecture [16], which employs NER-based entity recognition and translation. However, our approach extends beyond simple NLP parsing by incorporating context-aware intent refinement and multi-layered policy validation, improving the accuracy and adaptability of intent interpretation.

Several IBN frameworks incorporate AI-driven optimization. For example, IBN-Smart Distribution Grids Architecture [21] employs closed-loop optimization for network slicing in power grids, but suffers from latency concerns in real-time automation. Similarly, IBN-Connectivity & Cloud Architecture [28] applies AI/ML-based intent refinement, but faces interoperability issues in multi-cloud environments. Our approach builds on these AI-driven techniques by implementing real-time telemetry feedback loops, enabling adaptive intent refinement based on continuous network performance monitoring. This reduces the latency and inefficiencies found in traditional static intent execution models.

While OSDF-IBN Framework [8], [9] integrates hybrid proactive rule installation and reactive adjustments, it faces high complexity in policy reasoning. Similarly, IBN-Automation for IoT (VISCR) [23] uses graph-based conflict detection but suffers from high computational costs due to real-time policy validation. In contrast our framework mitigates these issues by allowing LLM to learn continuously from user feedback and refine intent processing over time. This adaptive learning mechanism reduces errors in intent interpretation, enhances natural language understanding, and supports multi-domain policy enforcement. Unlike static rule-based approaches, our LLM-driven intent processing evolves dynamically, improving accuracy and efficiency with each interaction.

7.3 Research Direction

The research direction proposed in this work has the potential to revolutionize the landscape of Intent-Based Networking (IBN) and Software-Defined Networking (SDN). Current research trends suggest a significant shift in both Northbound Interface (NBI) and Southbound Interface (SBI) research within the realm of Intent-Based Networking (IBN). Specifically, NBI-related research is increasingly gravitating toward the integration of Natural Language Processing (NLP) techniques to facilitate more intuitive and intelligent intent interpretation. On the other hand, SBI-related research has been progressing towards enhancing programmability within middleboxes, enabling more dynamic and fine-grained control over network policies and configurations. Building upon these trends, the approach proposed in this work seeks to merge these two complementary areas into a unified framework that combines NLP-driven IBN NLP-driven intent processing with programmability-enhanced middleboxes. The integration of these technologies introduces a number of possibilities that could lead to the next generation of intelligent, dynamic, and context-aware network management systems.

7.3.1 Immediate Impact and Emerging Trends in IBN and SDN

In the immediate future, the integration of NLP with IBN systems will likely enhance the ease of configuring networks by allowing users to express their intents in natural language, making network management more accessible to non-experts. This would

democratize the deployment and maintenance of network policies, eliminating the need for complex manual configurations. Coupled with the flexibility and programmability introduced by middleboxes, which provide more granular control over network traffic and policies, we can expect networks to become more adaptive and responsive to real-time conditions.

One key trend that will emerge from this research is the acceleration of SDN's evolution. With the addition of advanced intent-driven programmability, SDN systems will become even more dynamic, enabling networks to reconfigure themselves based on high-level user inputs. The next generation of SDN architectures will incorporate more intelligent decision-making processes, taking into account not just predefined network configurations but also real-time network conditions, user preferences, and intent specifications.

7.3.2 Long-Term Impact on Network Automation and Intelligence

Looking ahead, the long-term impact of this research will likely be seen in the development of self-optimizing networks capable of understanding and adapting to complex user intents. As NLP technology continues to mature, we can expect IBN systems to evolve into highly intelligent frameworks that can autonomously interpret natural language inputs, convert them into network policies, and dynamically adjust to network conditions. These advancements would make networks more user-friendly, reducing the dependency on human intervention and automating network management tasks that were previously time-consuming and error-prone.

In addition, the fusion of NLP with middlebox programmability could facilitate a new generation of network security protocols. By allowing network administrators to specify security policies in natural language and have them automatically implemented and enforced by programmable middleboxes, the proposed research could lead to more secure, flexible, and adaptive security frameworks. This shift would address current limitations in cybersecurity, such as the slow response times to emerging threats and the need for constant manual configuration updates.

7.3.3 Potential Challenges and Research Areas for the Next Decade

While the proposed research opens up a wealth of possibilities, it also raises several challenges that will need to be addressed in the coming years. One of the primary concerns will be the accuracy and reliability of NLP-driven intent processing. Although NLP has made significant strides, its application in network management must overcome challenges such as ambiguity in language, context understanding, and domain-specific terminology. Future research will need to focus on improving NLP algorithms to ensure that network intents are understood correctly and can be translated into precise, enforceable actions.

Another challenge lies in the programmability of middleboxes. While they offer flexibility and enhanced control, middleboxes can also introduce complexity in terms of network performance, scalability, and integration with existing network infrastructure. Over the next decade, research will likely focus on refining the programmability and performance of middleboxes, ensuring they can scale efficiently in large, dynamic networks without introducing bottlenecks.

7.3.4 What to Expect in the Coming Decades

In the coming decades, we can expect IBN systems to evolve into fully autonomous network management systems that can self-configure, self-heal, and optimize based on user-defined high-level intents. With the integration of machine learning (ML) and AI into IBN, these systems will become even more intelligent, learning from past interactions to predict and respond to future network requirements without direct human input. This transition from reactive to proactive management will make networks more resilient, efficient, and capable of handling dynamic workloads, traffic patterns, and user demands.

As SDN continues to mature alongside IBN, we can expect a greater degree of abstraction in network management. Network operators will no longer need to be

concerned with the minutiae of individual network components; instead, they will focus on defining high-level business goals and desired outcomes. The network will then automatically determine the best path to achieve these objectives, optimizing traffic flow, balancing loads, and ensuring security policies are enforced without manual intervention.

This shift will enable the creation of next-generation smart cities, enterprise networks, and large-scale cloud infrastructures, where networks can adapt on-the-fly to changing demands, user behaviors, and external events. The combination of NLP-driven IBN and programmable middleboxes will also foster an environment in which networks can seamlessly integrate with other intelligent systems, such as IoT devices, edge computing platforms, and AI-driven applications.

7.3.5 Conclusion

In conclusion, the proposed research into merging NLP-driven intent processing with middlebox programmability represents a critical step forward in the evolution of IBN and SDN. This research paves the way for a future where networks are more intelligent, adaptive, and user-friendly, allowing for the automated translation of user intents into dynamic, enforceable policies. As we look toward the next few decades, we can expect networks to become fully autonomous, capable of self-optimization, and able to handle the complexities of modern, multi-tenant environments without human intervention. This shift will undoubtedly transform the networking landscape, enabling new possibilities for automation, security, and performance at scale.

7.4 Summary

This chapter critically assesses the proposed NLP-driven intent-based networking (IBN) framework in the context of Software-Defined Networking (SDN) and compares it with existing architectures. The key advantages of our approach include improved intent interpretation, adaptive learning, and real-time telemetry-based refinement, overcoming many limitations of traditional frameworks. A comparative analysis highlights that while existing IBN frameworks support AI-driven

optimization and intent refinement, they often suffer from latency, scalability, and policy enforcement challenges. Our framework enhances intent processing through LLM-driven continuous learning, addressing these gaps by dynamically improving accuracy and efficiency. The research direction outlined explores the potential of NLP-driven IBN and programmable middleboxes, which could lead to more intelligent, adaptive, and autonomous network management systems. Immediate impacts include natural language-based network configuration, enhanced SDN programmability, and real-time adaptability. Long-term trends suggest the development of self-optimizing, fully autonomous networks that require minimal human intervention.

References

- [1] R. A. Addad, D. L. C. Dutra, M. Bagaa, T. Taleb, H. Flinck, and M. Namane, “Benchmarking the ONOS Intent Interfaces to Ease 5G Service Management,” in *2018 IEEE Global Communications Conference (GLOBECOM)*, Abu Dhabi, United Arab Emirates: IEEE, Dec. 2018, pp. 1–6. doi: 10.1109/GLOCOM.2018.8648078.
- [2] P. Amaral, J. Dinis, P. Pinto, L. Bernardo, J. Tavares, and H. S. Mamede, “Machine Learning in Software Defined Networks: Data collection and traffic classification,” in *2016 IEEE 24th International Conference on Network Protocols (ICNP)*, Singapore: IEEE, Nov. 2016, pp. 1–5. doi: 10.1109/ICNP.2016.7785327.
- [3] R. Beckett, R. Mahajan, T. Millstein, J. Padhye, and D. Walker, “Don’t Mind the Gap: Bridging Network-wide Objectives and Device-level Configurations,” in *Proceedings of the 2016 ACM SIGCOMM Conference*, Florianopolis Brazil: ACM, Aug. 2016, pp. 328–341. doi: 10.1145/2934872.2934909.
- [4] M. Beshley *et al.*, “Customer-Oriented Quality of Service Management Method for the Future Intent-Based Networking,” *Appl. Sci.*, vol. 10, no. 22, p. 8223, Nov. 2020, doi: 10.3390/app10228223.
- [5] J. Carner, A. Mestres, E. Alarcon, and A. Cabellos, “Machine learning-based network modeling: An artificial neural network model vs a theoretical inspired model,” in *2017 Ninth International Conference on Ubiquitous and Future Networks (ICUFN)*, Milan: IEEE, Jul. 2017, pp. 522–524. doi: 10.1109/ICUFN.2017.7993839.
- [6] I. Cinemre, K. Mehmood, K. Kravetska, and T. Mahmoodi, “Gradient-Based Optimization for Intent Conflict Resolution,” *Electronics*, vol. 13, no. 5, p. 864, Feb. 2024, doi: 10.3390/electronics13050864.
- [7] I. Cinmere, K. Mehmood, K. Kravetska, and T. Mahmoodi, “Direct-Conflict Resolution in Intent-Driven Autonomous Networks,” Jan. 16, 2024, *arXiv*: arXiv:2401.08341. doi: 10.48550/arXiv.2401.08341.
- [8] D. Comer and A. Rastegarnia, “OSDF: A Framework For Software Defined Network Programming,” in *2018 15th IEEE Annual Consumer Communications & Networking Conference (CCNC)*, Jan. 2018, pp. 1–4. doi: 10.1109/CCNC.2018.8319173.

- [9] D. Comer and A. Rastegarnia, “OSDF: An Intent-based Software Defined Network Programming Framework,” Jul. 06, 2018, *arXiv*: arXiv:1807.02205. doi: 10.48550/arXiv.1807.02205.
- [10] S. Donovan and N. Feamster, “Intentional Network Monitoring: Finding the Needle without Capturing the Haystack,” in *Proceedings of the 13th ACM Workshop on Hot Topics in Networks*, Los Angeles CA USA: ACM, Oct. 2014, pp. 1–7. doi: 10.1145/2670518.2673872.
- [11] K. Dzevaroska, J. Lin, A. Tizghadam, and A. Leon-Garcia, “LLM-based policy generation for intent-based management of applications,” in *2023 19th International Conference on Network and Service Management (CNSM)*, Oct. 2023, pp. 1–7. doi: 10.23919/CNSM59352.2023.10327837.
- [12] K. Dzevaroska, A. Tizghadam, and A. Leon-Garcia, “Intent Assurance using LLMs guided by Intent Drift,” Feb. 02, 2024, *arXiv*: arXiv:2402.00715. doi: 10.48550/arXiv.2402.00715.
- [13] A. Gulenko, M. Wallschlager, and O. Kao, “A Practical Implementation of In-Band Network Telemetry in Open vSwitch,” in *2018 IEEE 7th International Conference on Cloud Networking (CloudNet)*, Tokyo: IEEE, Oct. 2018, pp. 1–4. doi: 10.1109/CloudNet.2018.8549431.
- [14] Y. Han, J. Li, D. Hoang, J.-H. Yoo, and J. W.-K. Hong, “An Intent-based Network Virtualization Platform for SDN”.
- [15] J. Haxhibeqiri, P. H. Isolani, J. M. Marquez-Barja, I. Moerman, and J. Hoebeke, “In-Band Network Monitoring Technique to Support SDN-Based Wireless Networks,” *IEEE Trans. Netw. Serv. Manag.*, vol. 18, no. 1, pp. 627–641, Mar. 2021, doi: 10.1109/TNSM.2020.3044415.
- [16] A. S. Jacobs, R. J. Pfitscher, and R. H. Ribeiro, “Hey, Lumi! Using Natural Language for Intent-Based Network Management”.
- [17] A. S. Jacobs, R. J. Pfitscher, R. A. Ferreira, and L. Z. Granville, “Refining Network Intents for Self-Driving Networks,” *ACM SIGCOMM Comput. Commun. Rev.*, vol. 48, no. 5, pp. 55–63, Jan. 2019, doi: 10.1145/3310165.3310173.
- [18] C. Kim, A. Sivaraman, N. Katta, A. Bas, A. Dixit, and L. J. Wobker, “In-band Network Telemetry via Programmable Dataplanes”.
- [19] X. Lu, J. Chen, L. Lu, X. Huang, and X. Lu, “SDN routing optimization based on improved Reinforcement learning,” in *Proceedings of the 2020 International*

- Conference on Cyberspace Innovation of Advanced Technologies*, Guangzhou China: ACM, Dec. 2020, pp. 153–158. doi: 10.1145/3444370.3444563.
- [20] J. Mcnamara *et al.*, “NLP Powered Intent Based Network Management for Private 5G Networks,” *IEEE Access*, vol. 11, pp. 36642–36657, 2023, doi: 10.1109/ACCESS.2023.3265894.
 - [21] K. Mehmood, H. V. K. Mendis, K. Kravetska, and P. E. Heegaard, “Intent-based Network Management and Orchestration for Smart Distribution Grids,” May 12, 2021, *arXiv*: arXiv:2105.05594. doi: 10.48550/arXiv.2105.05594.
 - [22] A. Mekrache, A. Ksentini, and C. Verikoukis, “Intent-Based Management of Next-Generation Networks: an LLM-Centric Approach,” *IEEE Netw.*, vol. 38, no. 5, pp. 29–36, Sep. 2024, doi: 10.1109/MNET.2024.3420120.
 - [23] V. Nagendra, A. Bhattacharya, V. Yegneswaran, A. Rahmati, and S. Das, “An Intent-Based Automation Framework for Securing Dynamic Consumer IoT Infrastructures,” in *Proceedings of The Web Conference 2020*, Taipei Taiwan: ACM, Apr. 2020, pp. 1625–1636. doi: 10.1145/3366423.3380234.
 - [24] M.-T.-A. Nguyen, S. B. Souihi, H.-A. Tran, and S. Souihi, “When NLP meets SDN : an application to Global Internet eXchange Network,” in *ICC 2022 - IEEE International Conference on Communications*, Seoul, Korea, Republic of: IEEE, May 2022, pp. 2972–2977. doi: 10.1109/ICC45855.2022.9838633.
 - [25] M. Riftadi and F. Kuipers, “P4I/O: Intent-Based Networking with P4,” in *2019 IEEE Conference on Network Softwarization (NetSoft)*, Paris, France: IEEE, Jun. 2019, pp. 438–443. doi: 10.1109/NETSOFT.2019.8806662.
 - [26] B. K. Saha, D. Tandur, L. Haab, and L. Podleski, “Intent-based Networks: An Industrial Perspective,” in *Proceedings of the 1st International Workshop on Future Industrial Communication Networks*, New Delhi India: ACM, Oct. 2018, pp. 35–40. doi: 10.1145/3243318.3243324.
 - [27] R. Soulé *et al.*, “Merlin: A Language for Provisioning Network Resources,” in *Proceedings of the 10th ACM International on Conference on emerging Networking Experiments and Technologies*, Sydney Australia: ACM, Dec. 2014, pp. 213–226. doi: 10.1145/2674005.2674989.
 - [28] M. Toy, “Intent-based Networking for Connectivity and Cloud Services,” *Adv. Netw.*, vol. 9, no. 1, p. 19, 2021, doi: 10.11648/j.net.20210901.12.
 - [29] Y. Tsuzaki and Y. Okabe, “Reactive configuration updating for Intent-Based Networking,” in *2017 International Conference on Information Networking*

- (ICOIN), Da Nang, Vietnam: IEEE, 2017, pp. 97–102. doi: 10.1109/ICOIN.2017.7899484.
- [30] R. M. A. Ujjan, Z. Pervez, and K. Dahal, “Suspicious Traffic Detection in SDN with Collaborative Techniques of Snort and Deep Neural Networks,” in *2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, Exeter, United Kingdom: IEEE, Jun. 2018, pp. 915–920. doi: 10.1109/HPCC/SmartCity/DSS.2018.00152.
- [31] C. Xu, W. Zhuang, and H. Zhang, “A Deep-reinforcement Learning Approach for SDN Routing Optimization,” in *Proceedings of the 4th International Conference on Computer Science and Application Engineering*, Sanya China: ACM, Oct. 2020, pp. 1–5. doi: 10.1145/3424978.3425004.
- [32] Y. Zhang, X. Gong, Y. Hu, W. Wang, and X. Que, “SDNMP: Enabling SDN management using traditional NMS,” in *2015 IEEE International Conference on Communication Workshop (ICCW)*, London, United Kingdom: IEEE, Jun. 2015, pp. 357–362. doi: 10.1109/ICCW.2015.7247205.
- [33] “Intent Framework - ONOS - Wiki.” Accessed: Feb. 13, 2025. [Online]. Available: <https://wiki.onosproject.org/display/ONOS/Intent+Framework>

Appendix A

Name of student: D.P. Udugamasooriya (E/19/409)

In this research, my primary contributions focus on Named Entity Recognition (NER), intent conflict resolution, SDN optimization with Machine Learning (ML), and QoS optimization. These aspects are critical in ensuring that an NLP-driven Intent-Based Networking (IBN) system functions efficiently, accurately translates user intents, and optimally manages network resources within a Software-Defined Networking (SDN) environment.

Named Entity Recognition (NER) for Intent Processing

NER plays a crucial role in automatically identifying key networking parameters from user intents. My contribution involves fine-tuning NER models to extract domain-specific entities such as bandwidth requirements, latency constraints, security levels, and application types. This will be achieved through, custom dataset creation for SDN specific NER training and fine tuning transformer-based models to improve entity recognition accuracy.

Intent Conflict Resolution

In an IBN framework, a single user may define multiple network intents that could lead to conflicting policies. My contribution focuses on detecting and resolving such conflicts to ensure consistent and predictable network behavior. These conflicts may arise due to overlapping resource requests, contradictory security and performance policies and conflicts in QoS constraints. To address these issues, I will develop an intent conflict detection and resolution module that will use NLP-based reasoning to infer possible contradictions in a user's intent statements and classify conflicts based on predefined rules and machine learning-based intent analysis.

SDN optimization using Machine Learning

Optimizing SDN performance using Machine Learning (ML) is crucial for ensuring that intent-driven policies dynamically adapt to real-time network conditions. My contribution involves implementing ML algorithms for proactive network flow management to optimize traffic routing and using predictive models to anticipate network congestion and adjust SDN policies accordingly.

Appendix B

Name of student: G.G.N. Viduranga (E/19/413)

In this research, my primary contributions focus on template generation, intent suggestion mechanism, monitoring, and chat interface development. These components are essential for enhancing the NLP-driven Intent-Based Networking (IBN) system, ensuring accurate intent interpretation, seamless user interaction, and effective network management within a SDN environment.

Template Generation

I will create a dynamic template generation system that customizes SDN configurations based on user requirements. This approach allows the system to create flexible and customizable structures based on different networking scenarios.

Intent Suggestion Mechanism

I will develop an intent suggestion mechanism that helps users define precise network intents by analyzing past inputs and predefined policies. This will reduce ambiguity and ensure alignment with SDN requirements. By leveraging NLP techniques, the mechanism will enhance user experience by reducing ambiguity and guiding users toward accurate network configurations.

Monitoring

I will contribute to implementing a monitoring system that continuously collects real-time network performance data and feeds it into a Machine Learning (ML) model to determine optimal threshold values. This system will analyze key performance metrics, detect anomalies, and adjust network policies dynamically based on ML-driven insights.

Chat Interface Development

I will contribute to designing a chat interface that facilitates user interaction with the IBN framework in a conversational manner. This interface will allow users to input intents in natural language, receive system-generated recommendations, and get real-time feedback on network configurations.

Appendix C

Name of student: I.M.K.D.I. Wijerathna (E/19/446)

In this research, my primary contributions focus on deploying an emulated SDN environment, implementing a continuous improvement pipeline using user feedback, and deploying the NLP model. These contributions ensure the scalability, adaptability, and real-world applicability of the proposed NLP-driven Intent-Based Networking (IBN) architecture.

Emulated SDN Environment Deployment and Management

I will be responsible for setting up and managing the SDN test environment using Containernet, Mininet, and P4-enabled OpenFlow switches. This includes:

- Deploying and maintaining three different network topologies to continuously evaluate the scalability of our proposed framework.
- Automating deployment scripts to configure virtual network topologies and ensure continuous traffic flow.
- Integrating P4 and OpenFlow switches to explore programmable data-plane capabilities.
- Simulating basic attack scenarios to test network resilience and security policies.

Continuous Improvement Pipeline

I will develop a feedback-driven pipeline to enhance the ML engine and NLP model, using a versioned history store to track improvements and adapt to real-world user interactions dynamically.

NLP Model Deployment

I will handle NLP model deployment, including server setup, infrastructure configuration, and API integration, ensuring efficient hosting and real-time intent processing.

These contributions ensure the IBN system remains scalable, adaptive, and practical for real-world networking.