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

Dineth Udugamasooriya
Department of Computer Engineering
University of Peradeniya
Peradeniya, Sri Lanka
e19409@eng.pdn.ac.lk

Nipul Viduranga
Department of Computer Engineering
University of Peradeniya
Peradeniya, Sri Lanka
e19413@eng.pdn.ac.lk

Kalindu Wijerathna
Department of Computer Engineering
University of Peradeniya
Peradeniya, Sri Lanka
e19446@eng.pdn.ac.lk

Suneth Namal
Department of Computer Engineering
University of Peradeniya
Peradeniya, Sri Lanka
namal@eng.pdn.ac.lk

***Abstract*—This review paper examines the transformative role of Natural Language Processing (NLP) in advancing Intent-Based Networking (IBN) frameworks, particularly within Software-Defined Networks (SDNs). By connecting human-readable intents and business goals with machine-executable configurations, NLP-powered Northbound interfaces can streamline network management and enhance the user experience. Additionally, the integration of programmable elements in Southbound interfaces is crucial for achieving granular control over network operations. The paper outlines the latest advancements, challenges, and opportunities in IBN, emphasizing the need for incorporating NLP in Northbound interfaces and programmable components in Southbound interfaces to create more efficient, intelligent, and adaptable networking solutions. Through collaborative research and innovative approaches, the full potential of IBN can be realized, enabling networks that are not only responsive and efficient but also aligned with the evolving needs of businesses and users.**

***Index Terms*—Intent-Based Networking (IBN), Software-Defined Networking (SDN), Natural Language Processing (NLP), Machine Learning (ML), Intent Processing, Network Management, Network Orchestration, Large Language Models (LLMs), Intent Conflict Resolution, Network Monitoring, Network Telemetry**

I. 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. This shift towards intent-driven approaches has brought about significant improvements in automation, efficiency, and network management.

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).

The evolution of IBN frameworks has been facilitated by foundational SDN-based architectures, as outlined in various IETF RFC standards such as RFC 9316 [1] (which introduces the concept of intent in SDN) and RFC 9315 [2] (which focuses on standardized intent specification and processing) (IETF, 2023). These standards provide the necessary guidelines for building scalable and interoperable IBN solutions.

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 paper explores the current state of NLP-driven intent processing, highlighting key advancements, frameworks, and challenges, while also considering ongoing research in related domains such as intent conflict resolution, network monitoring, and telemetry. These efforts underscore the importance of developing more fully-complete NLP-driven IBN frameworks that can provide end-to-end solutions, thus encouraging further research in this area to realize the full potential of intelligent, autonomous networking systems in the future.

II. Related Works

This section 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

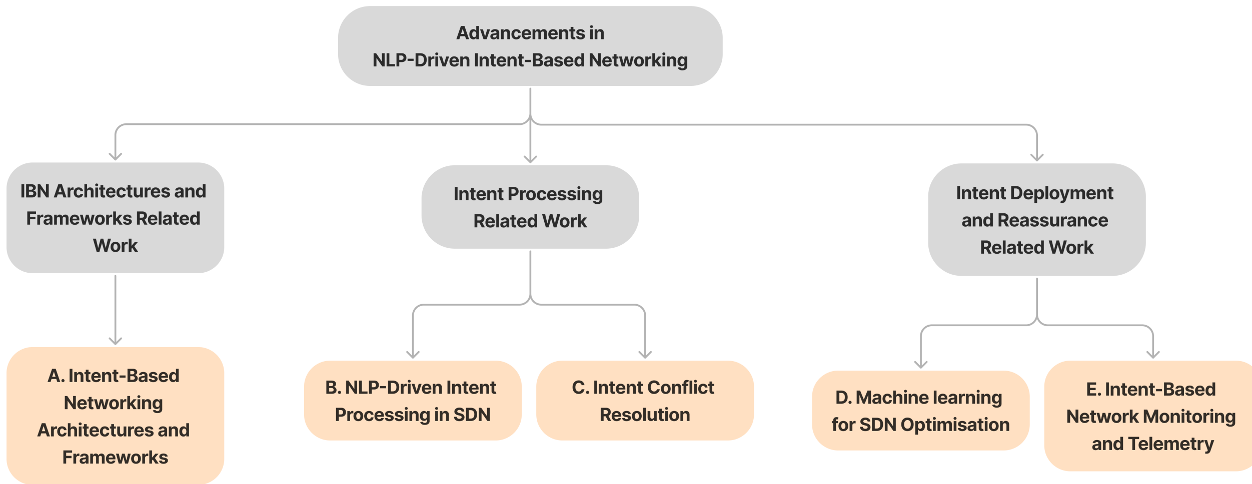

Fig. 1. Overview of Related Works in NLP-driven Intent-Based Networking.

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 1 provides an overview of the key areas explored in the literature.

A. 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 provides an overview of the categorized prominent IBN architectures chosen for this study.

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.

a) ONOS IBN framework [3], [4]: 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 "intents." 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 [3].

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 [4].

b) IBN-Virtualization platform architecture [5]: Extending ONOS, the IBN-Virtualization Platform introduces multi-tenancy support through network slicing and virtualization layers. 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.

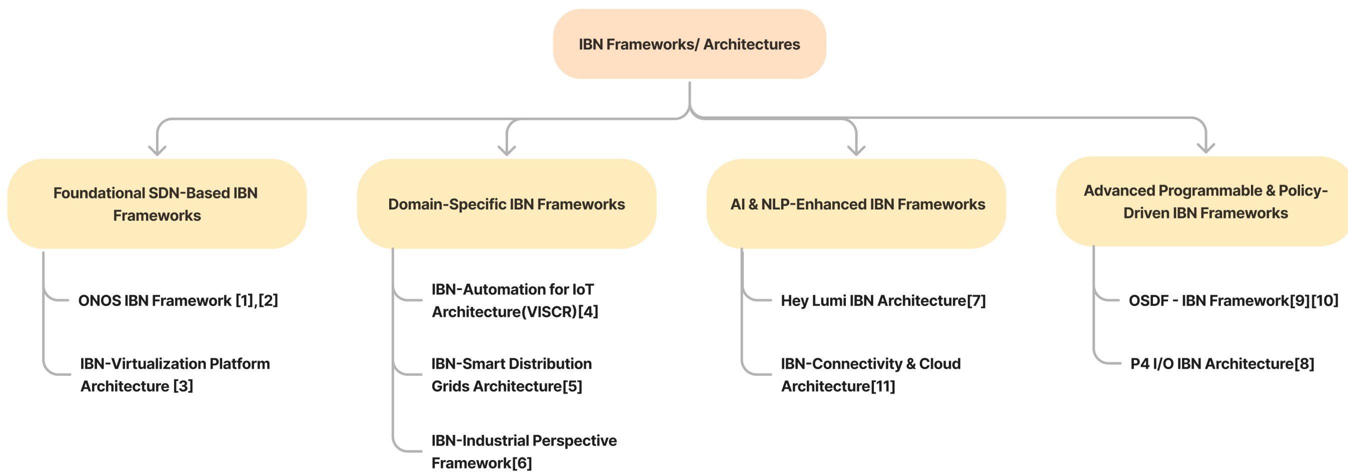

Fig. 2. Overview of the Categorization of Selected Prominent IBN Architectures/Frameworks.

TABLE I
COMPARATIVE ANALYSIS OF PROMINENT IBN FRAMEWORKS AND ARCHITECTURES

References	Framework/Architecture	Intent Specification	Processing Approach	Description	Challenges & Research Gaps
[3],[4]	ONOS IBN Framework	Structured high-level intents	Proactive & Reactive	Basic intent compilation, REST/CLI-based interfaces, ONOS-based architecture	Limited expressiveness, lacks AI-driven optimization, reactive adaptation is slow
[5]	IBN-Virtualization Platform Architecture	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
[6]	IBN-Automation for IoT Architecture (VISCR)	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
[7]	IBN-Smart Distribution Grids Architecture	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
[8]	IBN-Industrial Perspective Framework	High-Level Enterprise Policies	AI/ML-Based Optimization & Decision Making	AI-driven intent execution, Network-State Observer, Intent Prioritization	Lack of standardization for AI-driven intent processing, scalability issues in large enterprises
[9]	Hey Lumi IBN Architecture	NLP-based natural language intents	AI-driven Entity Recognition & Translation	NER-based intent parsing, Nile language compilation, Operator feedback refinement	Ambiguities in natural language interpretation, lacks robust policy enforcement and Southbound programmability
[13]	IBN-Connectivity & Cloud Architecture	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
[11],[12]	OSDF - IBN Framework	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
[10]	P4 I/O IBN Architecture	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) Domain-Specific IBN Frameworks: These frameworks tailor IBN capabilities to specific network environments, such as IoT security, smart grid automation, and industrial networks.

a) IBN-Automation for IoT architecture(VISCR) [6]: VISCR introduces a vendor-independent policy translation mechanism that uses graph modeling to address security and policy enforcement issues in IoT infrastructures. 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.

b) IBN-Smart distribution grids architecture [7]: This system 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:

i. **Intent Management Layer:** This layer is responsible for translating grid operator intents into service requirements, utilizing Natural Language Processing (NLP) to extract specific needs from operators.
ii. **Network Management Layer:** This layer manages the lifecycle of network slices, allocating resources based on the dynamic demands of the power grid.
iii. **Grid Infrastructure Layer:** This 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.

c) IBN-Industrial perspective framework [8]: Designed for enterprise and industrial networks, 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.

However, the lack of standardization for AI-driven intent processing poses challenges, particularly in terms of interoperability across different systems. Moreover, scalability limitations in large industrial deployments necessitate research into modular architectures that can adapt to varying operational scales without compromising performance.

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.

a) Hey Lumi IBN architecture [9]: This framework pioneers natural language-based intent parsing, allowing network operators to express requirements in human-readable form. 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 [10] and OSDF [11], [12], 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.

b) IBN-Connectivity & cloud architecture [13]: Targeting cloud-network integration, this AI-enhanced framework maps high-level service requirements to connectivity models using machine learning-based ranking algorithms. 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 I/O [10] and OSDF [11], [12], 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.

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).

a) OSDF - IBN framework [11], [12]: The Open Software Defined Framework (OSDF) introduces policy-driven SDN programming, supporting hybrid intent execution that combines proactive rule installation with reactive adaptation. 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 [14]*, 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.

Comparison with Merlin's [15] Conflict Resolution

Merlin was not considered a complete IBN framework in this paper 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.

- OSDF's conflict resolution uses policy analysis techniques to detect and resolve intent conflicts before they are installed, minimizing the risk of unintended rule overwrites. However, it does not support real-time negotiation or dynamic adjustment of intent constraints during runtime, which limits its flexibility in handling evolving network conditions.

- Merlin's Negotiator model incorporates a constraint-solving mechanism that enables network components to cooperate dynamically, adjusting policies in response to real-time conditions and interdependencies across multiple intents.

Thus, Merlin's approach is more adaptive and capable of handling real-time changes in network policies, but it is also computationally more complex. OSDF, on the other hand, prioritizes deterministic conflict resolution with lower runtime overhead but lacks the flexibility of a programmable negotiation layer, making it less effective in large-scale, multi-stakeholder networks where dynamic policy reconciliation is essential. Furthermore, while both OSDF's policy-driven enforcement and Merlin's compiler-driven approach translate intents into network rules before deployment, OSDF lacks Merlin's preemptive optimization capabilities, which could enhance efficiency in dynamic, large-scale deployments where policy conflicts frequently arise.

b) P4 I/O IBN architecture [10]: 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. Because of its adaptability, 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.

Intent-Based Networking (IBN) architectures have significantly evolved, progressing from foundational SDN intent translation systems such as ONOS and virtualization-based IBN, to more specialized applications in domain-specific automation for IoT, Smart Grids, and Industrial IBN. Advanced frameworks like Hey Lumi and Cloud & Connectivity IBN leverage AI to enhance the flexibility and intelligence of network management. Cutting-edge technologies, such as policy-driven SDN programming (e.g., OSDF) and high-performance programmable networks (e.g., P4 I/O IBN), have emerged, enabling the automated, real-time realization of network intent. As IBN architectures continue to advance, future innovations should focus on:

i. **Enhancing natural language-driven capabilities and reasoning** to facilitate more intuitive, user-friendly network management, enabling operators to define network intents using high-level, human-readable specifications without sacrificing control or complexity.

ii. **Optimizing intent-driven decision-making in real-time** leveraging AI to improve the network's responsiveness and adaptability to changing conditions, ensuring that decisions can be made swiftly and accurately based on current network states.

iii. **Standardizing cross-domain IBN** to enhance interoperability across diverse network environments, ensuring seamless communication between multiple network domains, such as cloud, edge, and data center networks, to support a unified, cohesive network operation.

iv. **Developing hybrid approaches that combine natural language-driven Northbound interfaces (NBIs) with programmable Southbound protocols.** These hybrid models would allow for scalable, intent-driven automation, offering granular control over network configurations while supporting complex network requirements and dynamic adaptations.

By focusing on these advancements, IBN frameworks can unlock greater scalability, flexibility, and efficiency, enabling the next generation of automated, intent-based network management.

B. 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 [16]. 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 [9]. 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 [17]. 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 [18]. 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 [19]. 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 [20]. 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.

C. 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 [21].

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) [22].

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 [23]. 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 [24]. 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.

D. 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

TABLE II
COMPARATIVE ANALYSIS OF NLP-BASED INTENT PROCESSING APPROACHES

Reference	Intent Processing Stages	NLP / ML Techniques Used	Intent Representation	Limitations
[16]	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
[9]	Information Extraction, Intent Assembly, Confirmation, Deployment	NER (Bi-LSTM + CRF for identifying and labeling entities from natural language intents)	Network Intent Language (Nile)	Lacks ambiguity detection and intent conflict resolution
[17]	Classification, Conversion, Confirmation, Implementation	LSTM-based Sequence-to-Sequence Model	Structured Network Intent Language (SNIL)	No intent optimizing mechanisms after deploying the intent
[18]	Decomposition, Translation Negotiation, Activation Assurance	LLM-based NLP processing	Infrastructure-Level Intents (ILIs)	Requires fine-tuned LLMs
[19]	Classification, Progressive Decomposition, Validation	Few-shot Learning, Feedback-driven Policy Generation	Structured Policy Tree	No intent drift detection
[20]	Intent Drift Detection, KPI Extraction, Assurance	Pre-trained LLMs, KPI-based error function	Formalized KPI-based intent	Requires historical KPI datasets

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 [25]. 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 [26]. 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 III
SUMMARY OF DRL-BASED ROUTING APPROACHES

Approach	Key Technique	State Representation	Action	Reward Function
DDPG [25]	Deep Deterministic Policy Gradient	Traffic Matrix (TM)	Adjust link weights	Throughput, Delay
DDPG-EREP [26]	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 [27]. 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

TABLE IV
ML-BASED NETWORK MODELING APPROACHES FOR DELAY ESTIMATION

Model Type	Technique	Assumptions	Accuracy	Scalability	Key Findings
M/M/1-based	Theoretically inspired quening model	Follows Jackson's network theory	Moderate	High	Less accurate under heavy traffic
ANN-based	Artificial Neural Network	Data driven, no fixed assumptions	High	Moderate	More adaptable to real-world conditions

in SDN-enabled environments. A proposed architecture for collecting OpenFlow-based traffic data enables efficient traffic classification using supervised ML algorithms [28].

E. Intent Based Network Monitoring and Telemetry

Software-Defined Networking (SDN) has transformed network management by introducing centralized control, programmability, and automation [29]. However, effective SDN management depends on real-time telemetry, which provides deep visibility into traffic flows, device performance, and policy enforcement [30]. 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 [29]. 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 [29]. 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 [29]. 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 [30].

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 [29]. 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 [29].

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 [30], [31], [32]. 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 [30]. 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 [30], [31], [32]. 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 [31]. 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 [33], [34]. 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 [30], [32], [35]. Additionally, security concerns arise, as telemetry metadata could be exploited for traffic analysis attacks, posing privacy and confidentiality risks [32], [35]. 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 [30], [32].

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 [33], [34]. Automated intent translation mechanisms, leveraging Natural Language Processing (NLP), will simplify the process of converting high-level network objectives into actionable policies [14]. Additionally, strengthening encryption and authentication protocols will secure telemetry data against potential cyber threats, ensuring that network monitoring does not introduce new vulnerabilities [35]. As networks continue to evolve, intent-based telemetry will play a vital role in creating intelligent, responsive, and self-adaptive infrastructures.

III. Conclusion

In this paper, 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.

References

[1] A. Clemm, L. Ciavaglia, L. Z. Granville, and J. Tantsura, "Intent-Based Networking - Concepts and Definitions," RFC Editor, RFC9315, Oct. 2022. doi: 10.17487/RFC9315.

[2] C. Li, O. Havel, A. Olariu, P. Martinez-Julia, J. Nobre, and D. Lopez, "Intent Classification," RFC Editor, RFC9316, Oct. 2022. doi: 10.17487/RFC9316.

[3] "Intent Framework - ONOS - Wiki," Onosproject.org, 2016. https://wiki.onosproject.org/display/ONOS/Intent+Framework
Accessed: Feb. 12, 2025. [Online].

[4] 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.

[5] Y. Han, J. Li, D. Hoang, J.-H. Yoo, and J. W.-K. Hong, "An Intent-based Network Virtualization Platform for SDN".

[6] 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.

[7] K. Mehmood, H. V. K. Mendis, K. Kralevska, 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.

[8] 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.

[9] A. S. Jacobs et al., "Hey, Lumi! Using Natural Language for Intent-Based Network Management".

[10] 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.

[11] 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.

[12] 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.

[13] 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.

[14] 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.

[15] 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.

[16] 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.

[17] 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.

[18] 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.

[19] K. Dzeparoska, 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.

[20] K. Dzeparoska, 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.

[21] I. Cinemre, K. Mehmood, K. Kralevska, and T. Mahmoodi, "Gradient-Based Optimization for Intent Conflict Resolution," Electronics, vol. 13, no. 5, p. 864, Feb. 2024, doi: 10.3390/electronics13050864.

[22] I. Cinmere, K. Mehmood, K. Kralevska, and T. Mahmoodi, "Direct-Conflict Resolution in Intent-Driven Autonomous Networks," Jan. 16, 2024, arXiv: arXiv:2401.08341. doi: 10.48550/arXiv.2401.08341.

[23] 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.

[24] 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.

[25] 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.

[26] 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.

[27] 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.

[28] 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.

[29] Y. Zhang, X. Gong, Y. Hu, W. Wang, and X. Que, "SD-NMP: 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.

[30] 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 Transactions on Network and Service Management*, vol. 18, no. 1, pp. 627–641, Mar. 2021. doi: 10.1109/TNSM.2020.3044415.

[31] C. Kim, A. Sivaraman, N. Katta, A. Bas, A. Dixit, and L. J. Wobker, "In-band Network Telemetry via Programmable Dataplanes."

[32] 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.

[33] M. Beshley et al., "Customer-Oriented Quality of Service Management Method for the Future Intent-Based Networking," *Applied Sciences*, vol. 10, no. 22, p. 8223, Nov. 2020. doi: 10.3390/app10228223.

[34] 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.

[35] 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.

Reviewed

Dr. Suneth Namal

Group members

Dineth Udugamasooriya
(E/19/409)

Nipul Viduranga
(E/19/413)

Kalindu Wijerathna
(E/19/446)