Development of an Intent-based Networking tool for Network Automation Using NLP

Mohit Rai
Habib University
Karachi, Pakistan.
mr06638@st.habib.edu.pk

Saad Fahim
Habib University
Karachi, Pakistan.
sf06583@st.habib.edu.pk

Shehryar Amin
Habib University
Karachi, Pakistan.
sa06442@st.habib.edu.pk

Iqra Siddiqui
Habib University
Karachi, Pakistan.
is06176@alumni.habib.edu.pk

Farhan Khan
Habib University
Karachi, Pakistan.
farhan.khan@sse.habib.edu.pk

Abdul Samad
Habib University
Karachi, Pakistan.
abdul.samad@sse.habib.edu.pk

***Abstract*—Modern-day networks require frequent and complex configuration, often prone to errors. Intent-based Networking (IBN) resolves this by allowing the administrator to specify the desired network behavior in plain English called "intent".The IBN system then automatically translates these intentions into specific network configurations and commands. This automated approach streamlines network management, reduces errors, and increases efficiency, simplifying network operations. This study investigates the Idea of IBN, utilizing Natural Language Processing (NLP) in the form of Large Language Models (LLM) to automate the process. Cisco Packet Tracer was utilized to create Network topologies, for three use cases namely; Point-to-Point (PPP), Access Control-Lists (ACL), and Internet Protocol Security (IPSec). Furthermore, the dataset was for these Topologies containing Intents and their corresponding Network Commands. We used three different models for training, which included; (a) OpenAI's Generative Pre-trained Transformer (GPT)-2 was trained using full finetuning, (b) Microsoft's Phi-2 was trained using Parameter Efficient Fine-tuning (PEFT), and (c) Microsoft's Phi-3 was trained using PEFT.**

***Index Terms*—Intent-based Networking, Natural Language Processing, Large Language Modelling, Parameter Efficient Fine-tuning**

I. Introduction

Modern-day Networking is the backbone of our digital world, however, it is still plagued by ever-increasing complexity: this complexity and traditional reliance on manual configuration and reconfiguration for the network infrastructure. Modern-day Networking relies heavily on technologies such as Network Functions Virtualization (NFV) and Software-Defines Networking (SDN), which are at the forefront of connecting devices. However, these still require the Network Administrator to input the Network commands for the configuration and reconfiguration of the Network, in case of the addition of new devices, changes in topology, and addition of new users.

Several issues with Software-Defined Networking such as High Captial Investment costs, lack of backward compatibility, limited multi-vendor support, complex migration process, and scalability reliability Issues, as detailed in [1], leave a vacuum for a new technology that can address these gaps and fill in the emerging technical requirements and complexities involved in the configuration of Networks on the daily basis by the Network Administrators.

Identify applicable funding agency here. If none, delete this.

Furthermore, shortcomings with Network Functions Virtualization include interoperability (integration of diverse systems) issues, complex management & orchestration of the traffic & function monitoring, programmability & distributed management, lack of efficient resource allocation & energy efficiency, security & trust issues and issues of dynamic scaling to meet the demand of users. In addition to these issues, explained in detail in [2], performance limitations remain at large without any definitive answer, across a wide range of use cases.

Intent-Based Networking (IBN) offers a compelling solution to simplify network management. Instead of requiring users to configure network settings manually, IBN allows users to express their desired network behavior in a straightforward, declarative manner [3]. It goes beyond simply making it possible for an operator to interact with the network using higher-layer abstractions as it enables operators to concentrate on their intended results, leaving the specifics to the IBN [4]. Our work outlined pertains to Intent Profiling and Intent Translation sections, mentioned in [3] as we have defined the intent in a user-friendly way and the network policy is determined the resulting Network commands, which are then given as the output from the LLM. We have tested this using different Models and training techniques.

The rest of the paper is organized in such as way that Section II presents related work to IBN and NLP followed by the Dataset Used to train the Models in Section III, followed by the design of the System in Section IV and this followed by the Performance Metrics in section V. Section VI provides the conclusion and possible future directions

II. Related Work

Recently, there has been a growing emphasis on Artificial Intelligence in Computer Networks [5], however, there are still few works in the space of allowing the user to utilize natural Language Processing to process their submitted high-level Intent. Moreover, there has been increasing emphasis on several definitions of Cisco, which defines IBN as a system where administrators specify desired network states as high-level intents automatically translated into configurations and actions, with continuous monitoring and adjustments [6]. Gartner describes IBN as automating network operations by translating business policies into configurations, using analytics and machine learning for predictive and corrective actions [7].

Network World sees IBN as a method where desired outcomes are defined by administrators, and the system dynamically adjusts the infrastructure using automation and AI [8]. TechTarget emphasizes IBN's capability to set network states through high-level policies, with automated implementation, performance monitoring, and adjustments [9]. The IETF defines IBN as managing network behavior through business policies, with continuous assessment and automatic enforcement of necessary changes [9]. Similarly, there have been several different methods of translating network commands from High-level intents to Network configuration commands.

Moreover, other Intent-related techniques include Template Blueprint Translation, which uses pre-defined templates to map high-level intents to specific network policies and configurations. These templates, often tailored for different network services, VNFs, SFCs, and QoS levels, are used for GUI-based intent expression [10]. Moreover, the mapping involves decomposing the intent into mapping operations that associate specific keywords with network descriptors, utilizing a matching process based on labels and key-value pairs to identify the best network policies that satisfy the intent [11]. In addition, this technique uses an iterative approach to refine the intent by adding more details at each step. It often uses a tree-like structure and leverages IBN models for mapping and translation [12]. Another technique considers the NSDs, and deployment templates with details about VNFs, SFCs, and their connections, which are used in this technique. NSDs are used by the intent translator to create NFV orchestrator configurations [13].

Furthermore, the keyword-based technique uses predefined keywords to trigger specific network actions or policies. It often relies on rule-based matching functions and tag classification. This process utilizes tokenization into keywords and a rule-based matching function for the intents [14]. Additionally, the semantics technique uses semantic relationships between intent elements to structure and translate the intent. It often relies on RDF graphs and OWL ontologies [15]. Layer-wise segregation of the Intent could take place, where the firs-layer stores the basic service requirement, second-layer makes the addition of the service parameters, and the final layer can comprehensively render the particular network tools required for the policy deployment, as explained in [16].

Likewise, the concept of state machine is also relevant for intent translation. In the concept of state machines, by treating intents as binary or boolean variables, the state machine context-based translation method—which uses Deterministic Finite Automata (DFA) in particular—analyses them. This involves using a model of three states, each in charge of handling a different intent clause, DFA converts intents into Event-Condition-Action (ECA) policies [17]. A policy database associates every clause with its matching state; a default policy state is generated if no match is found.

Additionally, machine learning has been increasingly utilized to translate user intents into Network Policies and Commands. Deep neural networks, such as the IDON framework, may identify service features and intentions in place of hand-established classes, which speeds up keyword-to-policy mapping. These models are kept in a policy database and trained using past intents [18]. Secondly, in this case, Reinforcement learning can be leveraged to identify the new policies when the intent does not match existing Network Policies. In addition, similar keywords can be clustered for syntactic and semantic analysis using machine learning techniques like word embedding and Convolutional Neural Networks (CNNs), which enhance intent interpretation [18]. Recurrent Neural Networks (RNNs) are used in sequence-to-sequence learning models to predict keywords and extract semantic correlations, resulting in vendor-agnostic language outputs. Finally, Strong Reinforcement Policy selection can be optimized through generative adversarial network learning, which makes proactive decisions about the optimal policy based on reward functions and deployed intents that are monitored.

Likewise, there have also been attempts to develop Intent translation languages, which represent the intent in a structured and often human-readable way. One such example is Nemo. Administrators can specify their intended network behavior using Network Modelling (NEMO), a language for expressing network intents, without having to provide technical specifics. It takes an organized method, defining intents in terms of "object," "operation," and "result." Converting vague intentions into specific configuration adjustments on network devices facilitates automation. To simplify network management and lower errors, an intent such as "Create a firewall to allow HTTP traffic from the client to the web server" can be expressed with ease. Future developments will support reactive intents and proactive intents, enabling dynamic configuration updates based on network events, explained in [19].

NILE (Network Intent Language) is a high-level, human-readable intent definition language that aims to bridge the gap between natural language instructions and complex network configurations [20]. It serves as an intermediary between operator requests and lower-level policy enforcers, such as SDN rules. Unlike traditional policy languages, which require specialized knowledge, NILE is more natural language-like, making it easier for network operators or even home users to understand and verify intended behavior. An intent such as "Please add a firewall to block access to Facebook for the marketing department" can be expressed in NILE using clear, concise syntax. This approach simplifies intent specification, improves portability by decoupling intent from deployment, and allows operators to provide feedback before actual policy implementation, reducing misconfigurations and increasing the overall efficiency of intent-based networking.

Similarly, Yet Another Next Generation (YANG) also serves as an important programming Language for Software-defined Networks and Intent-based Network Frameworks.A YANG model, for example, can specify a router's interfaces, IP addresses, and configurations, which an SDN controller or IBN system can then use to manage the router. This approach improves network configuration consistency and allows for automated changes and updates. The paper [21] emphasizes the importance of YANG models being compatible with IBN systems, as this allows for the translation of high-level intents into concrete network actions, paving the way for truly autonomous network management.

In addition to that, Yet Another Mark-up Language (YAML) also has the potential to implement Network Intent Translation [22]. This YAML structure can then be given to a special program that translates those instructions into specific commands that your network devices understand. This makes network automation much easier, as both humans and computers can easily understand the intent and the resulting actions. For example, An intent such as ”Prioritise traffic from the marketing department to the web server” could be represented in YAML by specifying the source (marketing department), destination (webserver), and action (prioritize). This YAML structure can then be used as input to a translation engine, which maps YAML elements to specific network commands, eventually generating the necessary configuration changes for the network devices.

Moreover, several attempts have been made to develop an Intent translation, which translates the Network Intents into deployable Network Policies and commands.LumiChatbot, [23]employs NILE, a human-readable intermediary language, to bridge the gap between natural language network intents and actionable policies. It extracts key elements from user requests, organizes them into a structured NILE intent, and then validates the intent with the user before compiling it into a network configuration language such as Merlin. This process ensures that the user's intent is accurately understood and implemented, making network management more intuitive and efficient.

Furthermore, The paper [24] introduces ETS-Chatbot, a framework that allows users to express network configuration intents using natural language. The system uses Rasa, an open-source NLP tool, to extract relevant keywords and entities from user input. These entities are then assigned to a predefined intent model, which specifies the network parameters and possible values. The intent model is tailored to the enterprise network's specific domain, ensuring that user requests are accurately interpreted. ETS-Chatbot combines NLP techniques with a structured intent model to translate natural language expressions into a format that can be understood and processed by the network, allowing for more intuitive and user-friendly network management.

In our project, we aim to leverage the power of Large language models (LLM) to translate the Network Intents into deployable Network commands. The approach utilizes a comprehensive dataset encompassing various networking protocols and scenarios and incorporates rigorous testing with a network simulation tool for real-world validation. Furthermore, it prioritizes efficiency by exploring techniques to reduce computational cost, making it a more practical and powerful solution for intent-based networking.

III. Dataset

The Dataset was curated using materials related to Cisco, including the Network Simulation tool; Cisco Packet Tracer, and Cisco Certified Network Professional Labs. Moving on, the dataset included Network intents and commands related to three use cases, Point-to-Point (PPP), Access Control Lists (ACLs), and Internet Protocol Security (IPSec) Protocol. The Dataset was generated by creating a fixed Network topology on Cisco Packet Tracer and then using the resources to generate the specific Network commands related to that Topology. Furthermore, the network commands and intent corresponding to those network commands were compiled in a CSV file, which was then further divided into train, validate and test data, fed to the LLM Models

IV. System Design

In this section, we delve into the fine-tuning process of Large Language Models (LLMs) and explain the specific techniques used to develop our systems. LLMs are trained on vast amounts of text data to predict the next word in a sentence. During this **pre-training**, the model is exposed to extensive text corpora and learns language patterns, grammar, and context by minimizing the difference between its predictions and the actual words in the text. This stage results in a generalized model with broad language understanding. A trained model can then be **finetuned** on task-specific data to utilize the model for the intended application. The finetuning process is shown in Figure 1.

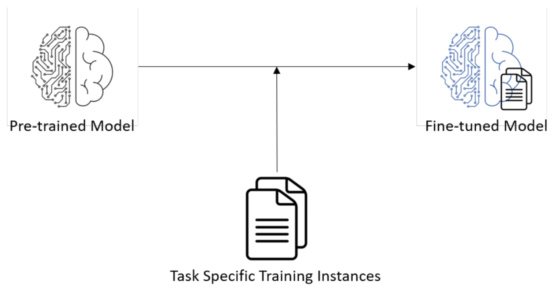


Fig. 1. Finetuning a pretrained model

The process of finetuning involves adjusting the weights of the pre-trained model to enhance the model's ability to generate relevant and accurate outputs for the given context. This can be done in two ways:

- **Full-Finetuning**: Full fine-tuning involves updating all the parameters of the pre-trained model using the new dataset. This approach can lead to significant improvements in performance for specific tasks as it allows the model to adapt comprehensively to the new data.
- **Parameter Efficient Finetuning**: Parameter-Efficient Fine-Tuning (PEFT) is an alternative to full fine-tuning that aims to reduce the computational and memory overhead associated with updating all model parameters. PEFT updates only a subset of the parameters, typically adding and fine-tuning smaller task-specific

modules while keeping the majority of the pre-trained model parameters frozen (see Figure 2). This method is particularly useful when computational resources are limited or when working with very large models

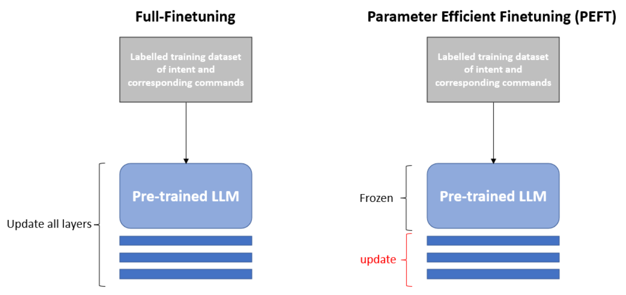

Full-Finetuning
Parameter Efficient Finetuning (PEFT)
Labelled training dataset of intent and corresponding commands
Labelled training dataset of intent and corresponding commands
Pre-trained LLM
Pre-trained LLM
Update all layers
Frozen
update

Fig. 2. Fullfinetuning Vs. PEFT

For our experiments, we used our dataset to perform full-finetuning on GPT-2 and utilized Microsoft's Phi-2 and Phi-3 models, both of which were trained using the QLoRA (Quantized Low-Rank Adaptation) PEFT technique. QLoRA combines two key techniques: quantization and low-rank adaptation.

- **Quantization** This process involves converting the model's weights from high-precision formats to lower-precision formats, such as from 32-bit floating-point to 8-bit integers. Quantization reduces the memory usage and computational requirements significantly without substantially affecting the model's accuracy.
- **Low-Rank Adaptation** In this technique, small, low-rank matrices are added to the model's layers. During fine-tuning, only these low-rank matrices are updated while the rest of the model's parameters remain mostly unchanged. Low-rank adaptation allows the model to learn task-specific adjustments efficiently. By keeping the majority of the model parameters frozen, this method not only reduces the number of parameters that need to be trained but also minimizes the risk of overfitting to the new data.

The following figure shows the overall design of the IBN tool.

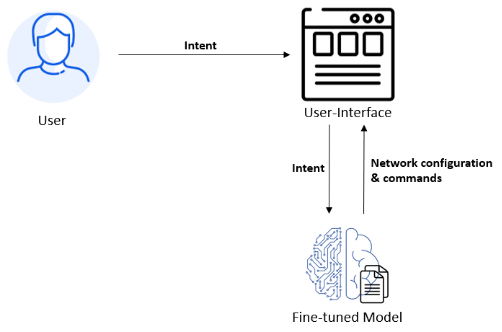


Fig. 3. System Design

V. Performance Analysis

In this section, we evaluate the performance of our models using two key evaluation metrics: BERTScore and BLEU score. These metrics provide a comprehensive view of the model's performance in generating accurate and relevant text.

A. Evaluation Metrics

BERTScore measures the similarity between predicted text and reference text using pre-trained contextual embeddings from BERT. It aligns words in the candidate and reference sentences and computes a similarity score. BERTScore is particularly useful for capturing semantic similarity. A good BERTScore is typically above 0.90, indicating high-quality text generation.

BLEU (Bilingual Evaluation Understudy) score is a precision-based metric that compares the overlap of n-grams between the generated text and reference text. BLEU scores range from 0 to 1, where higher scores indicate better performance. A BLEU score above 0.75 is generally considered good, showing that the model's outputs are closely aligned with the reference text.

B. Model Performance

The following table summarizes the BERTScore and BLEU score for each model:

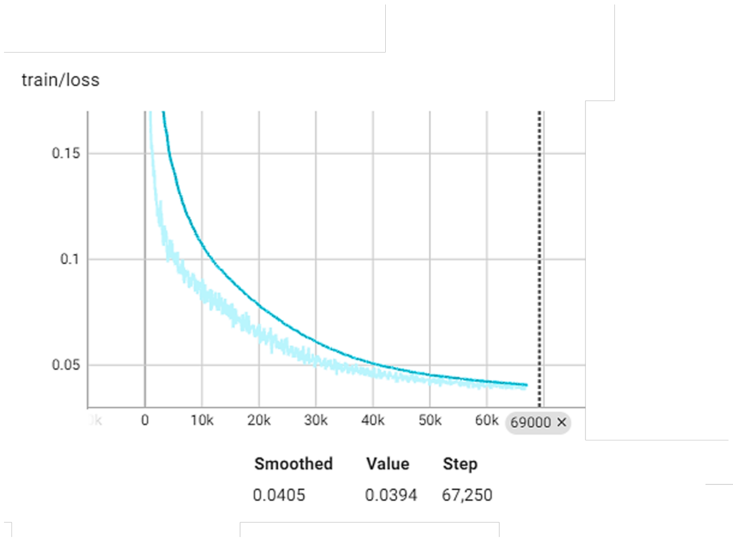
TABLE I
Model Performance Evaluation

Model	BERTScore	BLEU Score
GPT-2	0.92	0.79
Phi-2	0.95	0.84
Phi-3	0.96	0.86

Our results indicate that all models perform well, with Phi-3 achieving the highest BERTScore of 0.96 and BLEU score of 0.86, closely followed by Phi-2 and GPT-2. The high BERTScores across all models suggest that they generate text that is semantically similar to the reference, while the BLEU scores indicate good n-gram overlap, with Phi-3 and Phi-2 outperforming GPT-2.

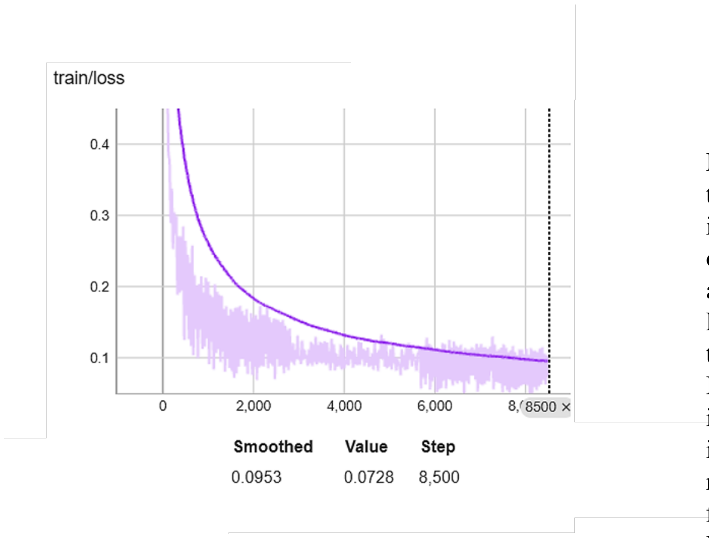
C. Training Loss Analysis

The training loss logged during the fine-tuning process provides insight into how well the model is learning. Figures 4, 5, and 6 show the loss curves for GPT-2, Phi-2, and Phi-3 respectively. The loss curves show that all models experience a steady decrease in loss over time, indicating effective learning.



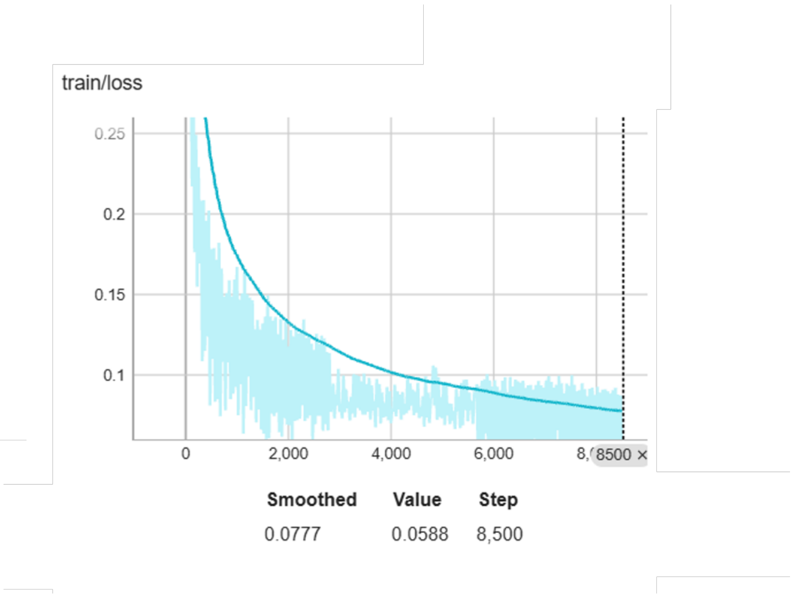
Smoothed	Value	Step
0.0405	0.0394	67,250

Fig. 4. Training Loss for GPT-2



Smoothed	Value	Step
0.0953	0.0728	8,500

Fig. 5. Training Loss for Phi-2



Smoothed	Value	Step
0.0777	0.0588	8,500

Fig. 6. Training Loss for Phi-3

The Phi-3 and Phi-2 models, which employ the QLoRA PEFT technique converge and produce better results in only 3 epochs, suggesting more efficient adaptation to the training data despite not being fully finetuned like GPT-2. GPT-2 achieves the best training loss but its compartively worse bleu and bert scores indicate a loss in the ability to generalize when introduced to slight variations of the intents provided during training alluding to some over-fitting to training data.

In comparing the models, several key points emerge:

- **Performance**: Phi-3 achieves the highest scores in both evaluation metrics, making it the most accurate model in our experiments.
- **Finetuning Techniques**: The use of QLoRA in Phi-2 and Phi-3 models demonstrates the effectiveness of PEFT, particularly with quantization and low-rank adaptation. This approach not only preserves model performance but also reduces training time since PEFT requires training for fewer epochs.

VI. Conclusion

This study looked into the viability of using Large Language Models (LLMs) for intent-based networking (IBN) automation, specifically their ability to translate high-level user intents into specific network configurations. Through rigorous testing on a comprehensive dataset and evaluation using BERTScore and BLEU score, the Phi-3 model, trained with the QLoRA PEFT technique, achieved the highest accuracy, highlighting the potential for parameter-efficient fine-tuning.The proposed IBN scheme is adaptable to various network architectures, including SDN and open-flow networks, and can be seamlessly integrated with enterprise networks via centralized configuration tools. Future research will develop specialized LLMs for networking, increase explainability and trust, and combine LLMs with other AI techniques to create more intuitive, efficient, and secure network management solutions.

References

[1] B. Sokappadu, A. Hardin, A. Mungur, and S. Armoogum, "Software defined networks: Issues and challenges," in *2019 Conference on Next Generation Computing Applications (NextComp)*, 2019, pp. 1–5.

[2] R. Mijumbi, J. Serrat, J. Gorricho, N. Bouten, F. De Turck, and R. Boutaba, "Network function virtualization: State-of-the-art and research challenges," *IEEE Communications Surveys Tutorials*, vol. 18, 09 2015.

[3] A. Leivadeas and M. Falkner, "A survey on intent-based networking," *Commun. Surveys Tuts.*, vol. 25, no. 1, p. 625–655, jan 2023. [Online]. Available: https://doi.org/10.1109/COMST.2022.3215919

[4] A. Clemm, L. Ciavaglia, L. Z. Granville, and J. Tantsura, "Intent-Based Networking - Concepts and Definitions," RFC 9315, Oct. 2022. [Online]. Available: https://www.rfc-editor.org/info/rfc9315

[5] H. Xu, "Application of artificial intelligence in computer network technology under the background of big data era," *Journal of Physics: Conference Series*, vol. 1550, p. 032033, 05 2020.

[6] Cisco, "Intent-based networking: Building a network that listens," https://www.cisco.com/c/en/us/solutions/intent-based-networking.html, 2024, accessed: 03-Jun-2024.

[7] Gartner, "Intent-based networking (ibn)," https://www.gartner.com/en/information-technology/glossary/intent-based-networking-ibn, 2024, accessed: 03-Jun-2024.

[8] N. World, "What is intent-based networking?" https://www.networkworld.com/article/3247840/what-is-intent-based-networking.html, 2024, accessed: 03-Jun-2024.

[9] TechTarget, "Intent-based networking (ibn)," https://www.techtarget.com/searchnetworking/definition/intent-based-networking-IBN, 2024, accessed: 03-Jun-2024.

[10] A. Leivadeas and M. Falkner, "A survey on intent-based networking," *Commun. Surveys Tuts.*, vol. 25, no. 1, p. 625–655, jan 2023. [Online]. Available: https://doi.org/10.1109/COMST.2022.3215919

[11] D. Chen, D. Gao, W. Quan, Q. Wang, G. Liu, and H. Zhang, "Promoting network automation for heterogeneous networks collaboration," in *2020 IEEE 92nd Vehicular Technology Conference (VTC2020-Fall)*, 2020, pp. 1–5.

[12] J. McNamara, L. Fallon, and E. Fallon, "A mechanism for intent driven adaptive policy decision making," 11 2020, pp. 1–3.

[13] D. Borsatti, W. Cerroni, G. Davoli, and F. Callegati, "Intent-based service function chaining on etsi nfv platforms," in *2019 10th International Conference on Networks of the Future (NoF)*, 2019, pp. 144–146.

[14] M. Toy, "Intent-based networking for connectivity and cloud services," *Advances in Networks*, vol. 9, p. 19, 03 2021.

[15] M. Kiran, E. Pouyoul, A. Mercian, B. Tierney, C. Guok, and I. Monga, "Enabling intent to configure scientific networks for high-performance demands," *Future Generation Computer Systems*, vol. 79, 04 2017.

[16] H. Mahtout, M. Kiran, A. Mercian, and B. Mohammed, "Using machine learning for intent-based provisioning in high-speed science networks," in *Proceedings of the 3rd International Workshop on Systems and Network Telemetry and Analytics*. New York, NY, USA: Association for Computing Machinery, 2020, p. 27–30. [Online]. Available: https://doi.org/10.1145/3391812.3396269

[17] J. Kim, E. Kim, J. Yang, J. Jeong, H. Kim, S. Hyun, H. Yang, J. Oh, Y. Kim, S. Hares, and L. Dunbar, "Ibcs: Intent-based cloud services for security applications," *IEEE Communications Magazine*, vol. 58, no. 4, pp. 45–51, 2020.

[18] K. Zhan, H. Yang, Q. Yao, X. Zhao, A. Yu, J. Zhang, and Y. Lee, "Intent defined optical network: Toward artificial intelligence-based optical network automation," 01 2020, p. T3J.6.

[19] Y. Tsuzaki and Y. Okabe, "Reactive configuration updating for intent-based networking," in *2017 International Conference on Information Networking (ICOIN)*, 2017, pp. 97–102.

[20] A. S. Jacobs, R. J. Pfitscher, R. A. Ferreira, and L. Z. Granville, "Refining network intents for self-driving networks," in *Proceedings of the Afternoon Workshop on Self-Driving Networks*, ser. SelfDN 2018. New York, NY, USA: Association for Computing Machinery, 2018, p. 15–21. [Online]. Available: https://doi.org/10.1145/3229584.3229590

[21] K. Mehmood, K. Kralevska, and D. Palma, "Intent-driven autonomous network and service management in future cellular networks: A structured literature review," *Computer Networks*, vol. 220, p. 109477, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128622005114

[22] R. Polli, E. Wilde, and E. Aro, "YAML Media Type," RFC 9512, Feb. 2024. [Online]. Available: https://www.rfc-editor.org/info/rfc9512

[23] A. S. Jacobs, R. J. Pfitscher, R. H. Ribeiro, R. A. Ferreira, L. Z. Granville, W. Willinger, and S. G. Rao, "Hey, lumi! using natural language for intent-based network management," in *Proceedings of the 2021 USENIX Annual Technical Conference*. USENIX, July 14–16 2021, open access to the Proceedings of the 2021 USENIX Annual Technical Conference is sponsored by USENIX. [Online]. Available: https://www.usenix.org/conference/atc21/presentation/jacobs

[24] E. El-Rif, A. Leivadeas, and M. Falkner, "Intent expression through natural language processing in an enterprise network," in *2023 IEEE 24th International Conference on High Performance Switching and Routing (HPSR)*, 2023, pp. 1–6.