Intent-Based 6G Management with Generative AI

Abdelkader Mekrache*†, Adlen Ksentini†, Ulrich Finger†
*Simula Metropolitan, Oslo, Norway
†Eurecom, Sophia Antipolis, France
abdelkader@simula.no, {ksentini, finger}@eurecom.fr

***Abstract*—The 6G era introduces unprecedented complexity in managing heterogeneous, large-scale, and dynamic network infrastructures. These challenges are addressed by the concept of Intent-Based Networking (IBN), which has emerged as a promising paradigm for autonomous network management, enabling users to express high-level objectives that are automatically translated and enforced by the network. However, current IBN solutions remain constrained by rigid structured specifications and limited assurance mechanisms. This paper presents an overview of PhD research leveraging Generative AI (GenAI), specifically Large Language Models (LLMs), to address three fundamental IBN challenges: (i) intent translation, (ii) intent assurance, and (iii) GenAI operations in IBN systems. We propose a set of novel frameworks validated on real 5G/6G testbeds. Most contributions are supported by demos, datasets, and open-source implementations, which are referenced in the design section of each contribution. This PhD positions GenAI as a key enabler for advancing autonomous and user-centric 6G.**

***Index Terms*—6G networks, GenAI, IBN.**

I. Introduction

6G networks represent the next generation beyond 5G, introducing very advanced resource-demanding use cases in terms of latency and throughput [1]. In this context, traditional network management frameworks are not effective as the infrastructure is becoming complex. Managing such a multi-domain environment manually or with legacy scripts is no longer feasible. Furthermore, the shift toward open architectures introduces additional management overhead in coordinating functions from multiple vendors [2]. IBN has emerged as a network paradigm aimed at enabling autonomous networking through the introduction of the concept of "intent", where users express their objectives at a high level by specifying "what" they want the network to achieve, rather than "how" to achieve it. This paradigm has recently become a major focus in both standardization bodies and industry initiatives targeting beyond 5G and 6G networks [3]. In current IBN systems, user intent is typically defined using human-readable formats such as JSON or YAML. These are designed to capture high-level service requirements, which are then translated into network configurations. A closed-loop assurance is employed to continuously monitor and ensure that the specified intent requirements are satisfied. Although IBN is labeled as "human-readable," a key limitation is that they still require users to understand specific syntax and domain-specific abstractions. As a result, users with limited network expertise often struggle to accurately express their intent without prior knowledge [4].

In this PhD work, we focus on enhancing existing IBN systems by adopting the simplest and most natural form of intent specification: natural language. This direction has become increasingly promising due to the rapid progress of Generative AI (GenAI), following the emergence of ChatGPT in 2022 [5]. These models have demonstrated strong capabilities in reasoning making them suitable candidates for natural language IBN. In this context, we formulate three primary Research Questions (RQs):

- RQ1: How can GenAI effectively decompose and translate imprecise natural language intents into valid, multi-domain network configurations (e.g., NSDs) across RAN, CN, and Cloud [4, 6, 7, 8]?
- RQ2: To what extent can GenAI provide trustworthy intent assurance by combining automated anomaly mitigation with intuitive, natural language explanations for the user [9, 10]?
- RQ3: How can the operational performance of GenAI in IBN, specifically latency, robustness, and domain accuracy, be optimized through the creation of specialized telecom-aware datasets and models [11]?

The structure of the paper is illustrated in Fig.1. Section II presents background on IBN and GenAI, providing the necessary foundations. The subsequent sections cover the three main research directions of this PhD: intent translation (Section III), intent assurance (Section IV), and GenAI-enabled IBN operations (Section V). Section VI discusses future directions, and Section VII concludes the paper.

II. Background

This section presents the background on IBN and GenAI.

A. Intent-Based Networking

IBN is an autonomous network management paradigm in which high-level objectives are expressed by users and automatically translated into low-level configurations [3]. An IBN system typically operates through five stages [3]. In this PhD, we focus specifically on two of them: intent translation, where high-level intents are converted into machine-executable policies; and intent assurance, where continuous monitoring ensures that network behavior remains aligned with the original intent [3]. IBN is further supported by standardization and industrial efforts that define its conceptual and architectural foundations, including IETF RFCs [12], TM Forum [13], and ETSI ZSM [14]. Moreover, Recent research and standardization efforts are increasingly exploring next-generation IBN systems that leverage natural language intents as a more intuitive and user-friendly form of expression, with GenAI emerging as a key enabler for this evolution.

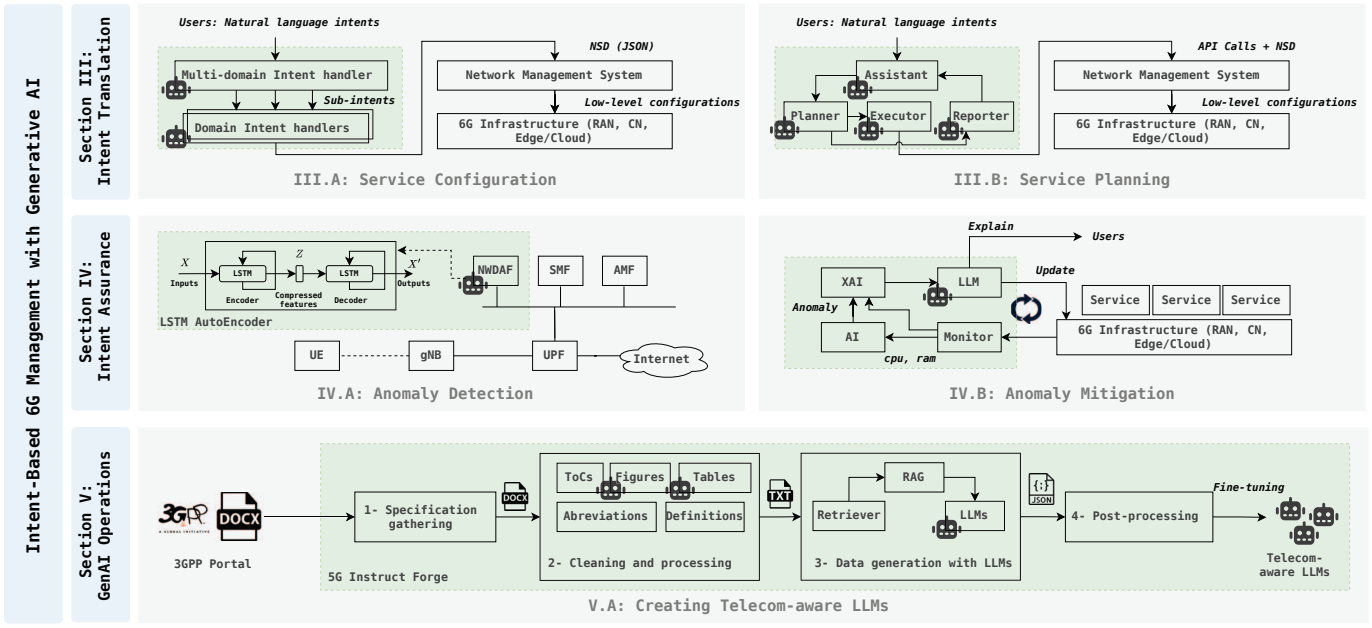

Fig. 1: PhD research directions. ■ indicates our contributions, while 🤖 represents a GenAI model.

B. Generative AI

GenAI, e.g. Large Language Models (LLMs), are AI systems capable of understanding and generating human-like text, and they are widely applied across a variety of tasks such as question answering, and code generation [5]. A broad ecosystem of LLMs currently exists, ranging from proprietary models such as GPT by OpenAI, Claude by Anthropic, to open-source alternatives such as Llama, Qwen. These models are typically built on transformer architectures [5]. Training such models requires large-scale compute infrastructure, and careful data curation to ensure diversity and quality. Once trained, LLMs are deployed for inference, where their performance depends on prompt design and context length [5]. Modern LLMs increasingly operate as interactive systems through prompting paradigms such as in-context learning, few-shot and Retrieval-Augmented Generation (RAG) [15]. More recently, LLM-based agents extend these capabilities by enabling multi-step reasoning, planning, and tool use, where models interact with external APIs and databases to execute complex tasks. Frameworks such as LangChain[1] and AutoGPT[2] enable LLMs to function as cognitive engines for decision-making tasks.

III. GenAI for Intent Translation

This section presents our contributions to intent translation. We begin with (i) service configuration, leveraging LLMs to generate low-level configurations. We then extend this in (ii) service planning, moving beyond configuration to plan the full intent workflow.

[1]https://www.langchain.com
[2]https://agpt.co

A. Service Configuration

1) Context: The problem we address here is that conventional IBN approaches rely on YAML/JSON formats to specify intents, which are not easily usable by users with limited domain knowledge. For example, ETSI defines a structure called the Network Service Descriptor (NSD), which is a JSON file that must be manually created by users and sent to the management system. Our goal is to generate this NSD using an LLM from natural language intents. This is powerful, as it enables users with limited expertise to manage networks using their spoken language without strict structural constraints. Indeed, prior work has explored this direction [16, 17], focusing on well-structured, unambiguous intents within a single domain (e.g., edge/cloud or CN). However, in real-world networks, intents often span multiple technological domains and must be translated into configurations across RAN, CN, edge/cloud simultaneously. Therefore, we propose an LLM-based approach that (i) decomposes the intent into sub-intents across different domains, (ii) translates these sub-intents into domain-specific configurations, and (iii) activates them.

2) Design: Our framework is composed of several components: (i) The multi-domain intent handler decomposes the intent into domain-specific sub-intents, each corresponding to a specific technological domain (RAN, CN, Edge/Cloud). This component is based on an LLM performing task decomposition via few-shot learning [18]. The few-shot examples are retrieved from a knowledge base. This enables the LLM to learn from relevant examples. A sentence transformer model is used for example retrieval to ensure that only semantically similar intents are included; (ii) The domain-intent handlers

translate the corresponding sub-intent into domain-specific configurations. We also perform syntactic and semantic validation of the generated configurations before activation. This step is critical to ensure that only correct configurations are propagated to lower layers; (iii) In both aforementioned components, we introduce a feedback loop that stores both the input/output in a knowledge base. This enables continuous learning, where future iterations can reuse past experiences when similar intents are encountered.

More information is available in [4, 6]. A demo[3] is also available.

3) Evaluation: We implemented the LLM system using the LangChain framework for LLM orchestration and ChromaDB[4] for vector storage and retrieval. We used CodeLlama[5] as the LLM engine and the MPNet v2 sentence transformer model[6] for semantic similarity and example retrieval. The Eurecom 5G facility [19] served as the underlying network management system. To evaluate performance, volunteer participants assessed the platform by creating 10 services and providing structured feedback. We adopted a rating-based evaluation approach: after each experiment, users rated both the intent decomposition and intent translation steps on a scale from 0 to 5. The results showed strong performance in intent decomposition, with an initial average rating of approximately 4.5. However, users initially reported lower satisfaction with intent translation, which required adjustments before generating NSDs compatible with the management system. Despite this, the average rating exceeded 3.5, indicating that only minor corrections were needed. Over time, the system improved through feedback, ultimately generating more accurate NSDs.

4) Limitation: The approach focuses only on intents aimed at generating network configurations. However, high-level intents may also involve other operations such as querying service status or deleting services. This requires a robust approach capable of generating and executing complete workflows per intent. We address this in the next contribution.

B. Service Planning

1) Context: Network management involves heterogeneous responsibilities in addition to service configuration. Some of the key functionalities include: (i) managing the lifecycle of 6G services (e.g., creation, activation, and deletion); (ii) managing infrastructure resources; and (iii) enabling observability and monitoring. Indeed, the previous subsection and existing research works [16, 17] have used LLM-based systems to enable service configuration; however, these approaches focus on a single task rather than a workflow of tasks. In more realistic scenarios, an intent should be translated into a workflow where the order of tasks is not predefined and must be inferred based on the intent content. In current management systems [19], tasks are executed via API calls, where each API corresponds to a specific operation. Consequently, the workflow we consider consists of defining the sequence of API calls, where each API may require structured inputs such as NSDs. Therefore, we design an LLM-based system that determines the workflow, generates the required configurations, and executes them to fulfill high-level intents. We used the ReAct framework [20], an agentic design pattern combining reasoning and acting, where reasoning corresponds to workflow planning and acting corresponds to executing it.

2) Design: We design our framework, called OSS-GPT, using a multi-agent architecture: (i) The assistant acts as the entry point of the system. It receives user intents and responds in natural language; (ii) The planner is responsible for generating the API workflow. Based on the API documentation and the intent, the planner generates the API calls workflow required to satisfy the intent; (iii) The executor executes the planned API calls sequentially using a set of tools. To perform API execution, it may need to generate structured inputs such as NSDs. For this purpose, the executor relies on a tool called the blueprint generator, which is an LLM specifically trained to generate NSDs. This latter is fine-tuned using the LoRA technique [21]. In this context, we used a dataset of intent–NSD pairs derived from evaluations of the previous contribution. This dataset is further augmented using few-shot prompting, and then used for fine-tuning; (iv) The final agent is the reporter, which communicates the outcomes to the assistant, which then forwards the response to the user.

More information are available in [7, 8]. A demo[7] is also available.

3) Evaluation: The OSS-GPT is implemented using LangGraph[8], and interacts with two LLMs: OpenAI's GPT-4[9], and the blueprint generator, deployed with Ollama[10].The blueprint generator was fine-tuned using Llama 3.2[11] as the base model with LoRa[21]. For training, we employed the Unsloth framework[12]. The Eurecom 5G facility [19] was used as the underlying network management system or OSS. To evaluate the quality of OSS-GPT, we prepared a dataset of intents generated by domain experts, ranging from simple to complex ones requiring multiple API calls (golden paths). For this purpose, we defined 8 golden paths (n). A golden path of i indicates that OSS-GPT needs to make at least i API calls. We used two metrics: (i) the number of API calls executed to fulfill an intent compared to the golden path; and (ii) the accuracy of successfully fulfilling intents for each golden path. For intents requiring fewer than 4 API calls, OSS-GPT achieved an accuracy close to 1.0, successfully fulfilling all intents. However, for intents with golden paths greater than 4, the accuracy gradually decreases as the complexity of the intents increases and more API calls are required. Notably, for golden path 7, OSS-GPT maintained an accuracy of 0.80, which is a strong performance given the complexity of the task.

[3]https://youtu.be/SDyBge8WMt0
[4]https://www.trychroma.com/
[5]https://huggingface.co/TheBloke/CodeLlama-34B-Instruct-GPTQ
[6]https://huggingface.co/sentence-transformers/all-mpnet-base-v2
[7]https://youtu.be/A1tTyHhyT80
[8]https://www.langchain.com/langgraph
[9]gpt-4o-2024-08-06
[10]https://ollama.com
[11]https://huggingface.co/meta-llama/Llama-3.2-3B-Instruct
[12]https://unsloth.ai

4) Limitation: The approach relies on closed-source LLMs, as they provided superior performance in planning. However, this dependency is being reduced, as open-source models are becoming increasingly powerful.

IV. GenAI for Intent Assurance

This section presents our contributions to the intent assurance phase. We begin with (i) anomaly detection using GenAI, and extend to (ii) anomaly mitigation, enabling autonomous resolution and closing the intent assurance loop.

A. Anomaly Detection

1) Context: Anomaly detection is the first critical stage of the intent assurance pipeline. In the standards, 3rd Generation Partnership Project highlights several anomaly detection use cases within the Network Data Analytics Function (NWDAF), including UE traffic anomaly detection. We investigate this use case, noting that the same methodology can be extended to other anomaly detection scenarios. NWDAF is a 3GPP-defined CN function that collects data from other network functions, and provides analytics, prediction, and anomaly detection services using AI [22]. At the time of this work, we identified a lack of open-source implementations of NWDAF, with most existing prototypes relying on simulations rather than real-world evaluations [23, 24]. To address this, we design a 3GPP-compliant NWDAF based on a microservices architecture enabling plug-and-play integration of use cases. We further implement an LSTM anomaly detection module and integrate it as a microservice within NWDAF.

2) Design: The proposed NWDAF follows a microservices-based architecture organized into three layers: Exposure, Monitoring, and Analytics. (i) Exposure acts as the entry point, providing 3GPP-compliant interfaces for external clients through NBI Gateway modules that handle both analytics queries and event subscriptions. Requests are routed either to real-time analytics services or subscription management services, which interact with the Analytics layer to compute and deliver results; (ii) Monitoring is responsible for continuous data collection from the network infrastructure. It aggregates telemetry from CN, Edge VIM resources, and RAN xApps [25], and stores them in a central database for further processing; (iii) Analytics hosts domain-specific engines mapped to NWDAF use cases, each applying either rule-based or AI-based models. In particular, the anomaly detection engine leverages an LSTM autoencoder trained on historical UE traffic patterns to compute abnormality probabilities. AI models are maintained in a dedicated repository and dynamically selected by the corresponding engines at runtime.

More information are available in [9]. A demo[13] is available. Both source code[14] and dataset [26] are available.

3) Evaluation: We implemented most of the NWDAF microservices using GoLang, while the anomaly detection engine was developed in Python. The setup includes an OAI-based gNB connected to a USRP B210 for radio transmission, and a second machine hosting both the OAI 5G CN and NWDAF. A commercial 5G UE (Quectel RM500Q-GL) was used for end-to-end validation. We used the Milano dataset [26], for model training originally collected by Telecom Italia over one year. The dataset was preprocessed into structured features including weekday, time, and traffic volume. Anomalies were synthetically injected by introducing long-duration traffic flows deviating from the original distribution to evaluate detection performance. We evaluate the LSTM autoencoder by measuring reconstruction loss convergence and its ability to model UE traffic patterns in the dataset. The model converges after approximately 10 epochs and successfully captures daily traffic dynamics. Using a detection threshold of 0.5, the model reliably identifies anomalous traffic patterns, with detections consistent with deviations from learned normal behavior, including short boundary effects induced by the sliding time window.

4) Limitation: The work does not address anomaly resolution. This remains an important step in the intent assurance. In the next contribution, we close the control loop in a different edge/cloud assurance problem.

B. Anomaly Mitigation

1) Context: Anomaly mitigation includes both anomaly detection and resolution. In this contribution, we focus on an intent deployed in the edge/cloud with CPU and RAM allocations and latency-related requirements. In this regard, anomaly detection occurs when the deployed intent violates latency requirements, and anomaly resolution consists of adjusting edge/cloud resources to mitigate this violation. There has been interesting work in this area, such as [27], where anomaly detection is performed using XGBoost [28], and resolution using SHAP [29] as an explainable AI (XAI) method. SHAP explains AI decisions by identifying the root features that contributed to the anomaly prediction, from which we can infer the root cause, whether CPU or RAM, and adjust resources accordingly. However, the main limitation of existing approaches is that they do not focus on natural language explanations, which are mandatory in intent assurance, as users need to understand what happens within the autonomous system. We address this limitation by extending [27] with an LLM-based agent that explains the anomaly detection and resolution process to users in natural language.

2) Design: Our pipeline design consists of three sequential steps: (i) In the first step, we use an XGBoost [28] to detect latency violations in deployed services. This model is trained on the cloud resource latency dataset presented in [27]; (ii) In the second step, we apply SHAP [28] to interpret the XGBoost predictions and identify the root cause of the latency violation, determining whether it is due to insufficient CPU, RAM, or both; (iii) In the third step, we use an LLM to generate both explanations and actionable outputs. This is implemented through a two-stage prompt engineering process: first, the model takes as input SHAP values and generates a detailed explanation of the root cause; second, this explanation is re-injected into the model to produce a structured JSON output

[13] https://www.youtube.com/watch?v=kI9GuJeW0es

[14] https://gitlab.eurecom.fr/oai/cn5g/oai-cn5g-nwdaf

describing the corrective action to be applied. This JSON format is machine-readable and can be directly consumed by the infrastructure orchestration layer to perform automated remediation, closing the loop without human intervention.

More information is available in [10]. A demo[15] and the training dataset[16] are available.

3) Evaluation: Llama2[17] was used as the LLM and deployed using text-generation-webui[18]. LangChain was used to implement the pipeline, while Kubernetes[19] was used to deploy the service intent. To trigger latency violations, we used ApacheBench[20] to generate synthetic workloads. The evaluation was conducted in a realistic experimental environment. The service intent was initially deployed with limited resources (0.25 CPU cores and 256 MB RAM) and subjected to synthetic workloads generated using ApacheBench, with varying levels of concurrent clients and request volumes to simulate different load intensities. The evaluation focuses on three aspects: latency violation detection under varying load conditions using XGBoost, root cause attribution using SHAP to distinguish between CPU and RAM related bottlenecks, and the effectiveness of Llama2 in generating both human-readable explanations and structured JSON-based remediation actions. Overall, the system demonstrates the ability to dynamically adapt resource allocations in response to workload variations, while generating accurate explanations. Minor inaccuracies in CPU allocation decisions are observed during specific stress transitions, highlighting limitations of generic LLMs.

4) Limitation: A key limitation of this framework is the use of a heavy LLM (70B) which was necessary to obtain sufficient reasoning capability, as smaller models were not yet reliable for this task. However, this introduces significant inference latency, which becomes a drawback in scenarios requiring low-latency decision-making in IBN assurance frameworks.

V. GenAI Operations in IBN

In previous sections, we identified limitations in GenAI in IBN. In this section, we therefore explore how to design GenAI approaches for IBN in order to improve these metrics.

A. Creating Telecom-aware LLMs

1) Context: In previous sections, we have highlighted the benefits of developing telecom-aware LLMs for IBN tasks. However, a key challenge lies in the requirement for high-quality datasets, which are scarce in the telecom domain [11]. In this context, 3GPP Technical Specifications (TSs) provide a rich source of domain knowledge, as they contain highly reliable and standardized networking information. However, their structural nature is not straightforward, as they are primarily available as document-based formats (e.g., DOCX). To address this challenge, we propose a framework that transforms 3GPP TSs into fine-tuning-ready datasets. This enables the adaptation of existing open-source LLMs through fine-tuning techniques to build specialized telecom-aware models. Such models are particularly beneficial for IBN tasks.

2) Design: Our proposed solution, 5G Instruct Forge, consists of a pipeline designed to transform 3GPP TSs into a fine-tuning-ready dataset: (i) In the first stage, specification gathering is performed either through automated download from the 3GPP portal[21]; (ii) In the second stage, cleaning and processing are applied to extract and structure relevant content from the TSs, including tables of contents, figures, tables, abbreviations, and definitions. Non-textual elements such as figures and tables are converted into textual representations using a Vision Language Model (VLM) and an LLM. The processed content is then concatenated into a unified textual corpus and embedded using text-embedding models; (iii) In the third stage, data generation is performed using LLMs, enabling the creation of multiple instruction-tuning tasks such as question answering, summarization, and text transformation in a RAG setting; (iv) Finally, a post-processing stage is applied to validate format consistency and remove invalid or incomplete samples, ensuring dataset quality for downstream fine-tuning of telecom-aware LLMs.

More information is available in [11]. Both the source code[22] and the training dataset[23] are available.

3) Evaluation: In Stage 2 of the pipeline, we use the MiniCPM[24] vision-language model (VLM) and GPT-3.5-turbo[25]. In Stage 3, we use two LLMs: GPT-3.5-turbo[26] and Llama3[27]. For RAG, we use ChromaDB[28]. We evaluate the proposed 5G Instruct Forge pipeline by generating the OAI Instruct dataset from 22 3GPP TSs and fine-tuning three open-source LLMs (Llama3[29], Solar[30], and Mistral[31]) using a freeze-tuning strategy in LLaMA Factory[32]. The resulting 5G-aware LLMs are evaluated using both multiple-choice and open-ended question-answering benchmarks derived from the OAI Instruct test set, with performance measured using semantic similarity metrics and generation latency. Overall, the results show that the generated dataset enables effective domain adaptation, improving 5G-specific knowledge acquisition while preserving general language capabilities.

B. Limitation

A key limitation is the static handling of evolving 3GPP TSs. When new specifications are released, the pipeline must be re-executed and the LLMs retrained. In addition, newer releases may override or replace previous information, creating challenges when legacy specifications must still be supported.

[15] https://youtu.be/1CoDNVWJcqA
[16] https://data.europa.eu/data/datasets/oai-zenodo-org-6907619
[17] TheBloke/Upstage-Llama-2-70B-instruct-v2-GPTQ
[18] https://github.com/oobabooga/text-generation-webui
[19] https://kubernetes.io
[20] https://httpd.apache.org/docs/2.4/programs/ab.html
[21] https://portal.3gpp.org/
[22] https://gitlab.eurecom.fr/netsoft/5g-instruct-forge
[23] https://huggingface.co/datasets/Netsoft/oai-instruct
[24] https://huggingface.co/openbmb/MiniCPM-V-2
[25] https://platform.openai.com/docs/models/gpt-3-5-turbo
[26] https://platform.openai.com/docs/models/gpt-3-5-turbo
[27] https://huggingface.co/meta-llama/Meta-Llama-3-70B-Instruct
[28] https://www.trychroma.com
[29] https://huggingface.co/meta-llama/Meta-Llama-3-8B-Instruct
[30] https://huggingface.co/upstage/SOLAR-10.7B-Instruct-v1.0
[31] https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.1
[32] https://github.com/hiyouga/LlamaFactory

VI. Futures directions

For intent translation, current approaches typically activate intents without any negotiation or conflict resolution, which remains a critical open challenge. When a user expresses an intent, the corresponding SLOs are derived; however, these SLOs may not be satisfiable due to limited infrastructure resources. In such cases, LLM agents could support intent negotiation by interacting with the user to resolve conflicts. This may include proposing alternative resource allocations, revising service plans, or adjusting priorities. Intent negotiation and conflict resolution therefore remain largely unexplored and represent a promising research direction.

For intent assurance, while LLM-based decision-making improves accuracy, it introduces significant drawbacks in terms of inference latency and energy consumption. A promising direction is the development of smaller, specialized language models (SLMs) specialized for decision-making tasks, which could reduce both energy footprint and inference time. However, this raises challenges related to the lack of high-quality, domain-specific telecom datasets, which are difficult to construct but essential for further progress in this direction.

For GenAI operations, 3GPP releases differ significantly in terms of knowledge, while backward compatibility is required when designing SLMs. We envision a modular architecture composed of release-specific specialized LLMs, where each model is trained on a given 3GPP release. A routing mechanism would dynamically select the most appropriate model based on the user query and the relevant release context. This approach represents a promising step toward version-aware telecom foundation models.

VII. Conclusion

This paper summarizes PhD work on exploring GenAI for 6G IBN. We first present the background, then focus on three main research directions: (i) intent translation, covering LLM-based methods for configuring and planning 6G services from natural language intents; (ii) intent assurance, addressing anomaly detection and resolution using GenAI; and (iii) GenAI operations, investigating specialized telecom GenAI approaches to improve latency and accuracy. Finally, we outline key future directions, including intent negotiation, SLMs, and modular release-aware architectures for 3GPP-aligned foundation models.

References

[1] Marco Giordani et al. "Toward 6G networks: Use cases and technologies". In: *IEEE communications magazine* 58.3 (2020), pp. 55–61.

[2] Estefania Coronado et al. "Zero touch management: A survey of network automation solutions for 5G and 6G networks". In: *IEEE Communications Surveys & Tutorials* 24.4 (2022), pp. 2535–2578.

[3] Aris Leivadeas and Matthias Falkner. "A survey on intent-based networking". In: *IEEE Communications Surveys & Tutorials* 25.1 (2022), pp. 625–655.

[4] Abdelkader Mekrache, Adlen Ksentini, and Christos Verikoukis. "Intent-based management of next-generation networks: An LLM-centric approach". In: *Ieee Network* 38.5 (2024), pp. 29–36.

[5] Yupeng Chang et al. "A survey on evaluation of large language models". In: *ACM transactions on intelligent systems and technology* 15.3 (2024), pp. 1–45.

[6] Abdelkader Mekrache and Adlen Ksentini. "LLM-enabled intent-driven service configuration for next generation networks". In: *2024 IEEE 10th International Conference on Network Softwarization (NetSoft)*. IEEE. 2024, pp. 253–257.

[7] Abdelkader Mekrache, Adlen Ksentini, and Christos Verikoukis. "Oss-gpt: An llm-powered intent-driven operations support system for 6g networks". In: *2025 IEEE 11th International Conference on Network Softwarization (NetSoft)*. IEEE. 2025, pp. 155–163.

[8] Abdelkader Mekrache, Adlen Ksentini, and Christos Verikoukis. "DMO-GPT: An Intent-Driven Framework for Distributed 6G Management and Orchestration". In: *IEEE Communications Magazine* (2025).

[9] Abdelkader Mekrache, Karim Boutiba, and Adlen Ksentini. "Combining network data analytics function and machine learning for abnormal traffic detection in beyond 5g". In: *GLOBECOM 2023-2023 IEEE Global Communications Conference*. IEEE. 2023, pp. 1204–1209.

[10] Abdelkader Mekrache et al. "On combining XAI and LLMs for trustworthy zero-touch network and service management in 6G". In: *IEEE Communications Magazine* 63.4 (2024), pp. 154–160.

[11] Azzedine Idir Ait Said et al. "5g instruct forge: An advanced data engineering pipeline for making llms learn 5g". In: *IEEE Transactions on Cognitive Communications and Networking* 11.2 (2024), pp. 974–986.

[12] Alexander Clemm et al. *Intent-Based Networking - Concepts and Definitions*. 2022. URL: https://www.rfc-editor.org/info/rfc9315.

[13] TM Forum. *Intent Toolkit*. URL: https://www.tmforum.org/toolkits/intent/.

[14] ETSI ISG ZSM. *Zero-touch Network and Service Management (ZSM); Intent-driven Autonomous Networks; Generic Aspects*. Tech. rep. GR ZSM 011 V2.1.1. ETSI, 2024. URL: https://www.etsi.org/deliver/etsi_gr/ZSM/001_099/011/02.01.01_60/gr_ZSM011v020101p.pdf.

[15] Yao-Yang Liu et al. "A comprehensive taxonomy of prompt engineering techniques for large language models". In: *Frontiers of Computer Science* 20.3 (2026), p. 2003601.

[16] Jieyu Lin et al. "AppleSeed: Intent-Based Multi-Domain Infrastructure Management via Few-Shot Learning". In: *2023 IEEE 9th International Conference on Network Softwarization (NetSoft)*. IEEE. 2023, pp. 539–544.

[17] Jingyu Wang et al. "Network Meets ChatGPT: Intent Autonomous Management, Control and Operation". In: *Journal of Communications and Information Networks* 8.3 (2023), pp. 239–255.

[18] Tom Brown et al. "Language models are few-shot learners". In: *Advances in neural information processing systems* 33 (2020), pp. 1877–1901.

[19] Sagar Arora et al. "A 5G Facility for Trialing and Testing Vertical Services and Applications". In: *IEEE Internet of Things Magazine* (2022).

[20] Shunyu Yao et al. "React: Synergizing reasoning and acting in language models". In: *arXiv preprint arXiv:2210.03629* (2022).

[21] Edward J Hu et al. "Lora: Low-rank adaptation of large language models." In: *Iclr* 1.2 (2022), p. 3.

[22] 3GPP. *Technical Specification Group Core Network and Terminals; 5G System; Network Data Analytics Services;* tech. rep. 29.520. Version 17.10.0. 2023.

[23] Dimitrios Michael Manias, Ali Chouman, and Abdallah Shami. "An NWDAF Approach to 5G Core Network Signaling Traffic: Analysis and Characterization". In: *GLOBECOM 2022 - 2022 IEEE Global Communications Conference*. 2022.

[24] Yachao Yuan et al. "Insight of Anomaly Detection with NWDAF in 5G". In: *2022 International Conference on Computer, Information and Telecommunication Systems (CITS)*. 2022.

[25] Robert Schmidt, Mikel Irazabal, and Navid Nikaein. "FlexRIC: An SDK for next-Generation SD-RANs". In: CoNEXT '21. 2021.

[26] Gianni Barlacchi et al. "A multi-source dataset of urban life in the city of Milan and the Province of Trentino". In: *Scientific Data* 2 (Oct. 2015), p. 150055. DOI: 10.1038/sdata.2015.55.

[27] Mohamed Mekki et al. "XAI-Enabled Fine Granular Vertical Resources Autoscaler". In: *2023 IEEE 9th International Conference on Network Softwarization (NetSoft)*. IEEE. 2023, pp. 161–169.

[28] Tianqi Chen and Carlos Guestrin. "Xgboost: A scalable tree boosting system". In: *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*. 2016, pp. 785–794.

[29] Scott M Lundberg and Su-In Lee. "A unified approach to interpreting model predictions". In: *Advances in neural information processing systems* 30 (2017).