GNN-ATIVE: An AI-native, Graph-based Orchestrator for Next-Generation Wireless Networks

Varun Gowtham*†, Osman Tugay Basaran‡†, Abhishek Dandekar‡†, Hanif Kukkalli§, Florian Schreiner*, Marius Corici*†, Julius Schulz-Zander‡, Falko Dressler†, Thomas Bauschert§, Slawomir Stanczak‡†, Thomas Magedanz*†

*Fraunhofer FOKUS, Berlin, Germany
†Technische Universität Berlin, Germany
‡Fraunhofer HHI, Berlin, Germany
§Technische Universität Chemnitz, Germany

***Abstract*—Traditional rule-based or static management approaches struggle to cope with the dynamic, multi-layered nature of 5G/6G networks, creating a strong motivation for AI-native solutions – management systems built from the ground up with artificial intelligence – to enable autonomous, real-time network control. In this work, we introduce GNN-ATIVE, an AI-native orchestration framework that leverages Graph Neural Networks (GNNs) and knowledge graphs (KGs) in a unified graph-based paradigm for network management. GNN-ATIVE uses a semantic knowledge graph to represent the network's state and context, employing standard ontologies to ensure consistency and interoperability. Building on this foundation, we design Knowledge Graph enabled Generative Pretrained Transformer (KG-GPT), a novel graph-to-graph Transformer model that performs knowledge-driven reasoning on the KG. KG ingests the structured network state (nodes, links, and attributes) and infers optimal configurations or management actions, serving as a high-level decision engine for the orchestrator. We implement and evaluate GNN-ATIVE on an Optical Transport Network (OTN) testbed using real network components. The results demonstrate that GNN-ATIVE can effectively manage OTN resources and adapt to network changes while achieving low-latency inference for decision making.**

***Index Terms*—AI-native, 6G, Artificial Intelligence, GNN, Intent-based Networking**

I. Introduction

Traditional orchestration systems and Operations Support Systems (OSS) rely heavily on predefined rules and human expertise, which struggle to adapt to fast-changing conditions. As networks evolve toward 6G, there is a clear need for smarter, autonomous management solutions. Artificial Intelligence (AI)-native network orchestration has emerged as a vision in which AI is deeply embedded into the management plane from the outset. Such an approach promises zero-touch automation and real-time adaptability, which are crucial for handling the complexity of modern telecom environments [1].

A key challenge in realizing AI-driven orchestration is how to represent and reason about the network's state and policies in a machine-understandable way. Graph Neural Networks (GNNs) have emerged as powerful AI tools for learning and decision-making on graph-structured data. In the context of network management, GNNs can learn representations (embeddings) of network elements and their inter-dependencies, aiding tasks such as traffic prediction, anomaly detection, or resource optimization. GNNs naturally capture the relational structure of networks, allowing models to generalize across different topologies and scenarios. However, using GNNs to orchestrate a live network requires an architecture that can handle both localized decisions and global reasoning across the entire system. So it is important that an effective AI-native orchestrator should combine data-driven learning with knowledge-driven reasoning, motivating a multi-tier design where AI models operate at different levels of abstraction and timescales, all informed by a shared network knowledge base. Graph-based models have gained traction as a natural fit for network representation: the topology of physical and virtual resources forms a graph structure, and additional information (such as performance metrics, configurations, or service dependencies) can be captured as attributes or related nodes. By using a Knowledge Graph (KG) to model the network, it can incorporate rich semantic information through standard ontologies and well-

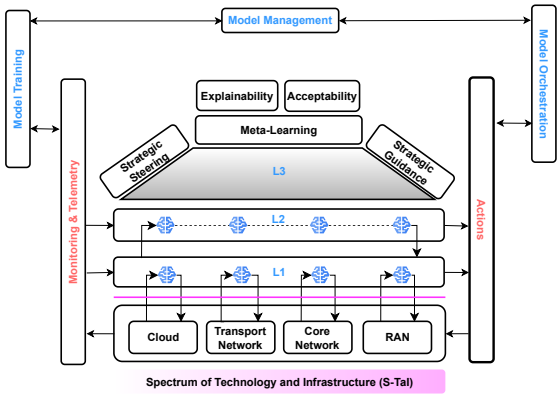

Fig. 1: GNN-ATIVE High-Level Framework

defined relationships, providing a unified view that spans devices, links, and services. This knowledge representation allows the orchestrator to leverage formal semantics – for example, describing resources in the Resource Description Framework (RDF) and querying the network state with SPARQL[2] – enabling advanced reasoning while also ensuring interoperability with standardized data models.

In this work, we present Graph Neural Network based AI-Native Orchestrator (GNN-ATIVE), a novel framework for orchestrating next-generation wireless networks using a graph-based, AI-centric approach. GNN-ATIVE treats the network and its operational data as a graph and builds a layered AI engine on top of this representation. Figure 1 shows the framework proposed in this article. L1, L2 and L3 hold dedicated roles based on the classification of their assigned tasks. L1 consists of models of low complexity, trained to capture simple policies and addressing a targeted control of the underlying infrastructure. Figure 1 lays out the relevant domains of telecommunication networks (cloud, transport network, core network and Radio Access Network (RAN)) and related technologies horizontally resulting in a Spectrum of Technology and Infrastructure (S-TaI). Parts of the S-TaI may then be optimized by first level of handlers, depicted by the layer L1, interacting directly with the components of S-TaI in the physical world. For example, an implementation of the L1 model presented later, maps an incoming maximum aggregate ingress data rate into a scheduled capacity, specifically interacting with a transponder in Optical Transport Network (OTN). The L2 models present the next abstraction level. A set of L1 models come under the purview of a L2 model, such that the L2 model manages a specific set of problems and furthermore can interact with other L2 models to manage a domain. Finally the L3 model, at the highest abstraction level is built using a reasoning model, for example using the transformer architecture as detailed later. The monitoring and actions provide necessary interfaces to receive data from and convey decisions to the S-TaI respectively. This approach, clearly separates the AI-native system composed of L1, L2, L3 models and the non-AI-native S-TaI elements. The framework also captures the importance of model training, management and orchestration by incorporating these facilities around the hierarchical arrangement of the proposed layers. In summary, this paper makes the following key contributions:

- We introduce the GNN-ATIVE framework, which to the best of our knowledge is the first orchestration system that is AI-native and graph-based.
- We develop Knowledge Graph enabled Generative Pretrained Transformer (KG-GPT), a novel graph-to-graph Transformer model tailored for knowledge graph reasoning in network management.
- We present a real-world evaluation using an OTN testbed, demonstrating that GNN-ATIVE meets low-latency inference requirements and effectively orchestrates network configurations in this setting.

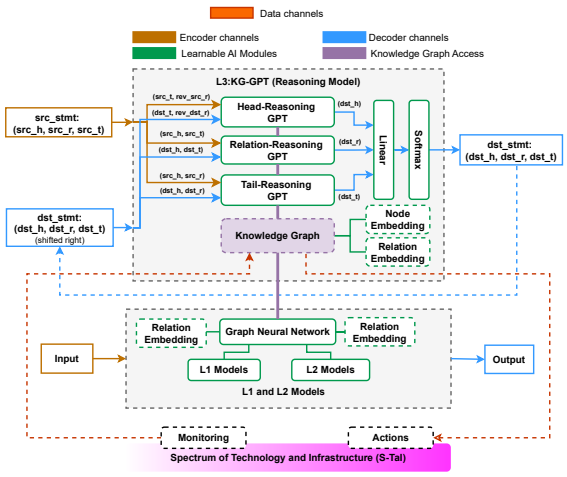

Fig. 2: GNN-ATIVE: L3:KG-GPT and GNN

II. AI-Native Design of Next-Gen Mobile Networks

With the emergence of Software-Defined Network (SDN) separating the control plane and data plane for the management of networks, and the subsequent evolution of Network Function Virtualization (NFV), disaggregation of networks has driven the network functionalities towards softwarization. Following this trend, the 3rd Generation Partnership Project (3GPP)'s Service Based Architecture (SBA) architecture and the recent arrival of Open RAN Alliance (O-RAN) have contributed to further disaggregation, in effect exposing Close Control Loops (CCLs) in a distributed software-based architecture that were previously in a monolithic structure. Managing the large manifest of software blocks becomes challenging for the Network Management System (NMS) to achieve agreed Service Level Agreement (SLA) and even more so when meeting energy and performance requirements. The limitation arises with traditional Policy Based Management and Control (PBMC) having to additionally take the burden of anticipating deviations and accounting for them with one or a combination of policies during the design of the management system. Since NFV advocates the distribution of responsibilities into dedicated Virtual Network Functions (VNFs), each VNF thus implements its policy based on the task assigned and exposes its administrative Application Programming Interface (API).

Due to the heterogeneous and diverse nature of the networks, a NMS has to adapt to diverse data streams. Two broad directions emerge [3] when applying AI in Network Management (NM): 1) exclusive models developed on dedicated data streams and use cases - although they are highly effective in addressing the use case, they are inflexible; 2) inclusive models such as transformer, for

which data has to be converted into an intermediate representation (for example, natural language in case of a Large Language Model (LLM)), - although they are highly flexible to host multiple reasoning channels, they could hallucinate when addressing specific problems [4]. Apart from natural language, KG has been explored as a possible bridge between inclusive and exclusive models [5]. KG as an intermediate data representation layer captures the network knowledge and state, such that the AI-native models can train more effectively thanks to its structure and non-ambiguous representation. Furthermore, following standardized ontology, KG can be effectively exchanged between stakeholders [6]. GNN has been widely explored in NM in the form of preparing exclusive models, some examples include: 1) RAN optimization [7, 8], 2) management of virtual infrastructures [9, 10], 3) network slicing [11], and 4) SDN network routing assistance [12, 13]. Deep learning and GNN have been leveraged to process topological and situation-specific graphs for the problem of VNF placement [9, 10]. GraphNET was built using GNN for assisting the SDN controller to predict better routes [12]. Graphs were also used to collect and summarize the information related to topology, traffic patterns and routing for the purpose of predicting delay and jitter, along with a combination of topologies [13, 14]. The TM Forum's IG 1230 set of technical guidelines [15] advocates the architecture for autonomous NMS with distributed intelligence, along with suggesting the Intent-based Networking (IBN) framework [16] as an enabler for achieving autonomous networks. The IG 1253 relies on the so-called Intent Management Function (IMF) as the atomic building blocks holding abstracted knowledge, logic and actions; here each IMF is responsible for one or many CCLs. Although the framework is intuitive, it lacks a reference implementation due to the heterogeneity of the network, requiring a collaborative environment between exclusive and inclusive models. GNN-ATIVE is a first look on this collaborative environment between L1, L2 and L3 models bridged by the KG. The advantage of management through distributed CCL has been well documented [12, 17]. In a similar direction, the European Telecommunications Standards Institute (ETSI)'s Zero touch network and Service Management (ZSM) proposes a conceptual architecture for zero-touch service management, also lacking a reference implementation [18].

III. The GNN-ATIVE Framework Design and Implementation

Figure 2 depicts a possible implementation of the framework shown in Figure 1. AI models have been shaped into two categories (exclusive and inclusive), specifically when applied to telecommunication networks as explained in Section II. This implementation leverages a combination of GNNs and KG. A graph consists of a set of nodes interconnected by a single type of edge. Similarly, a KG consists of edges of multiple types. Using standardized

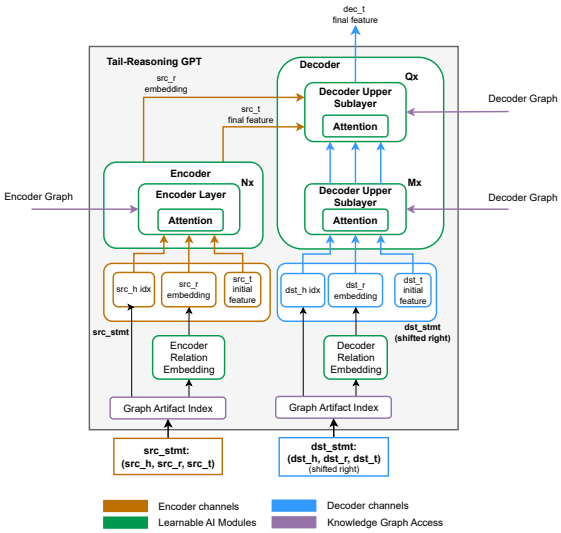

Fig. 3: L3:KG-GPT Tail Reasoning Sub-model

ontology, it's possible to express knowledge that is both human and machine-readable. Extending further, the KG with the help of an agreed-upon ontology can act as a digital twin of knowledge, that is unambiguously interoperable/exchangeable between stakeholders. The KG layer shown in Figure 2 forms the data layer, simultaneously capturing the network state and the knowledge. Above the KG layer, the models named L3:KG-GPT (using a transformer architecture) and L1:SCHED-CAP (using stacked linear layers) show examples of L3 and L1 models respectively constructed using a dedicated GNN pipeline. As this article focuses on an initial feasibilty analysis of the GNN-ATIVE framework, the discussion of L2 model is kept out of scope for the sake of brevity. Together, the L3 and L1 models act in coordination to manage an OTN detailed in Section IV.

```
1  @prefix ex1: <http://www.example.org/1/> .
2  @prefix ex2: <http://www.example.org/2> .
3
4  ex1:head1 ex2:relation ex1:tail1 .
5  ex1:head2 ex2:relation ex2:tail2 .
```

Listing 1: Example KG in RDF Turtle Syntax

The KG is represented using RDF. Listing 1 shows an example of the KG in turtle syntax [19]. The turtle syntax allows representing an RDF graph in a compact textual form. The example KG shown in Listing 1 is composed of two statements. Each statement is a triple with three elements- a head, a relation and a tail. Each element in the triple is represented using Universal Resource Identifier (URI) whose prefixes have been declared in the first half of the example. The KG can also be viewed as a sequence of statements. The generation of the KG was made simpler using the Domain Specific Language (DSL)-to-RDF toolkit. The toolkit sequentially processes DSL expressions provided by the user and consults the provided ontology to construct a semantically structured KG, simultaneously

verifying the integrity of the KG. The role of S-TaI, the monitoring and the actions blocks have already been explained in Figure 1. The KG can be hosted on the disk or on the so-called *triple-store* database, with which the monitoring and action agents can populate and query the information using SPARQL queries, respectively, from the KG. Through SPARQL queries; create, update, read and delete operations are possible on graph statements. Using dedicated GNN pipelines, it's possible to construct L1, L2 and L3 models. Drawing parallels to tokens in natural language, using GNN on KG results in embedding vectors generated for each node and edge present in the KG. The generated embedding vector is rich in features capturing the semantics and context expressed in the KG, thus easing the process of training L1 and L2 models. The L3:KG-GPT model simplifies the Graph2Graph problem to Statement2Statment problem; however, instead of a sequence of tokens in the case of LLM, the L3:KG-GPT transacts using sequence of statements. It ingests and generates KG, providing the reasoning capability. L3:KG-GPT further performs three reasoning tasks, namely: head, relation and tail reasoning. For example, the tail reasoning task in L3:KG-GPT, accepts the head and relation of the source statement and generates the tail of the destination statement. A similar generalization is possible for relation and head reasoning tasks required to generate a complete destination statement. The KG-GPT thus holds three separate sub-models for each of the reasoning tasks. Figure 3 shows the model for tail reasoning following the transformer architecture [20].

This architecture has been extended from "Knowformer" [21] using its attention blocks and reasoning mechanisms, to align with the transformer encoder-decoder architecture. Knowformer was originally proposed as an encoder-only architecture for entity or relation reasoning tasks, specifically exercised on KG. It exploits the Relational Message Passing Neural Network (RMPNN) to facilitate the attention mechanism, processed on the entire KG. The tail reasoning sub-model extracts the indices of the head and the relation of the source statement. The encoder accepts the embedding of the relation and the index of the head, fed to the query-key attention layer. The tail is sent as a zero-initialized context tensor into the encoder. Along with the index of the head and context tensor of the tail, the encoder executes a value attention layer. The KG is made available to the encoder, which is then applied N number of times on the input. The outgoing context tensor of the tail of the source statement is then fed to the decoder's upper sublayer along with the embedding of the relation in the source statement to perform the query key attention. Similarly, the decoder lower sub-layer, the head and relation from the previously generated destination statement is ingested to generate the context tensor of the tail of the destination statement, which is then forwarded to the upper sublayer to finally compute the score of the tail of the destination statement. The decoder sub-layers could also be applied multiple times.

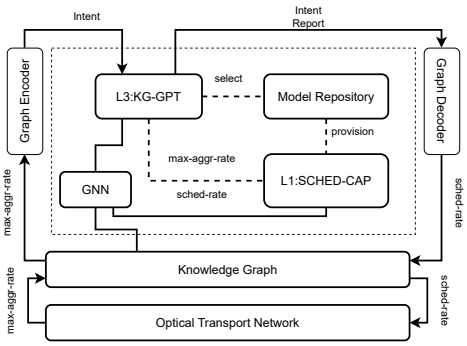

Fig. 4: Model Inference Procedure

With a similar model applied for head and relation reasoning, L3:KG-GPT generates the destination statement for each source statement. Given a KG as a source graph, L3:KG-GPT ingests a sequence of statements from the source KG to generate a sequence of destination statements resulting in a destination KG.

IV. Use-Case Evaluation

Figure 4 presents the experimental setup showing the proof of concept of the proposed AI-native framework with GNN. The system interacts with and controls an OTN, consisting of an aggregation switch, aggregating the traffic from the RAN and serving as the gateway to the OTN, followed by programmable Optical Transponders (OTs) and Reconfigurable Optical Add-Drop Multiplexers (ROADMs), which represent the OTN layer itself [22, 23]. The aggregation switch reports regularly through telemetry the maximum aggregated ingress traffic data rate (max-aggr-rate) to a dedicated telemetry framework [24], which then feeds these max traffic rate values into the KG. Finally, a dedicated NETCONF-module communicating directly with the KG-module, gets the newly computed, scheduled capacities (sched-cap) periodically from the KG, reconfiguring the OT when needed. Both processes use SPARQL query. To provision the scheduled capacity, the OT has its optical link-rate reconfigured in increments of 100 Gbps, offering only two possible capacities of either 100 Gbps or 200 Gbps. This scheme can also be successfully utilized on software-configurable coherent multi-carrier transceivers [25], where the capacity provisioning is achieved with a 25 Gbps capacity granularity.

The KG-GPT was constructed using the KG as explained in Section III. A Graph Encoder periodically querying the max-aggr-rate from the KG to construct the source graph as shown in Listing 2. The input graph depicts an intent expressed using the ontology suggested by [16, 26]. The graph consists of an intent, holding an expectation along with an associated expectation parameter. The value of max-aggr-rate is converted into an entity in the graph, i.e. out of the possible 400 values in increments of 0.5

from 0 Gbps to 200 Gbps, the graph contains 400 nodes associated with each possible max-aggr-rate value.

KG-GPT ingests the statements sequentially, until receiving the icm:target toco:otnScheduledBandwidthService. At this point, GraphGPT delegates the inference of the sched-cap to a L1 model selected via model repository. The L1 model (L1:SCHED-CAP) is trained to map an incoming max-aggr-rate to one of the sched-cap value of [100, 200] Gbps for the OT, or [0, 25, 50, 75, 100, 125, 150, 175, 200] Gbps for the Coherent Multi-Carrier Transceiver (CMCT). Similar to max-aggr-rate, the sched-cap are also available as nodes in the KG. Using the convolutional encoder ConvE GNN [27] on the KG, the embedding vectors of the max-aggr-rate were extracted to train the L1 model, which is a simple model consisting of 3 linear layers. The model accepts an embedding vector of dimension 200 of the max-aggr-rate to generate a one-hot vector of dimension 2 and 8 to signify the sched-cap, the results are shown in Figure 4 (both for OT and CMCT). The obtained result is then populated into the intent-report as shown in Listing 3, in accordance with the ontology suggested by TM Forum [26] along with the inferred sched-cap and forwarded to Graph Decoder. The Graph Decoder then updates the value associated with the transponder in the KG to be subsequently sent asynchronously to the OT. Figure 5 shows the inference time (from left to right) of the head reasoning submodel (L3:KG-GPT:HR-SM), the KG-GPT (in milli seconds), the KG-GPT (in milli seconds) and the L1 model (in micro seconds). The KG was constructed using the DSL-to-RDF toolkit as explained in Section III, consisting of 2142 nodes, 136 relations and 9090 statements (triples).

```
@prefix toco: <http://purl.org/toco/> .
@prefix icm: <https://www.tmforum.org/2020/07/
    ↪ IntentCommonModel/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .

toco:intentStartSubject toco:hasStartStatement toco:
    ↪ intentStartObject .
toco:ex1Intent rdf:type icm:Intent .
toco:ex1Intent icm:hasExpectation toco:
    ↪ otnDeliveryExpectation .
toco:otnDeliveryExpectation icm:target toco:
    ↪ otnScheduledBandwidthService .
toco:otnDeliveryExpectation icm:params toco:
    ↪ otnMaxAggrBandwidthParam .
toco:otnMaxAggrBandwidthParam icm:targetDescription toco:
    ↪ MaxAggrIngressRate_26_point0_Gbps .
toco:intentEndSubject toco:hasEndStatement toco:
    ↪ intentEndObject
```

Listing 2: Source Graph

```
@prefix toco: <http://purl.org/toco/> .
@prefix icm: <https://www.tmforum.org/2020/07/
    ↪ IntentCommonModel/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .

```

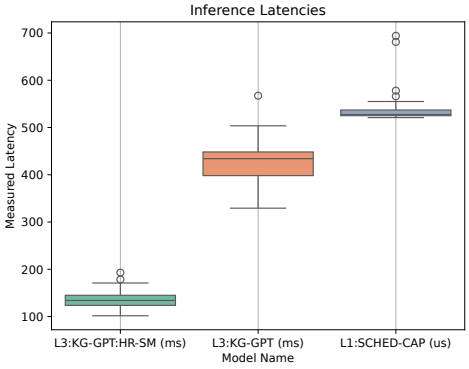

Fig. 5: Model Latency

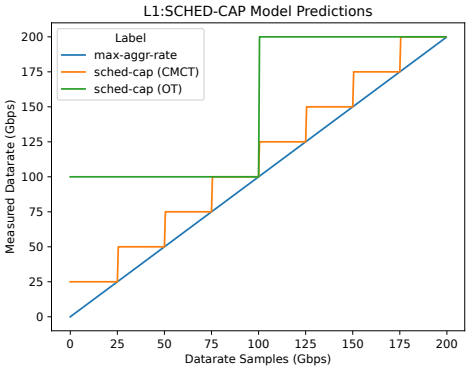

Fig. 6: L1:SCHED-CAP Model Predictions

```
toco:intentReportStartSubject toco:hasStartStatement toco:
    ↪ intentReportStartObject .
toco:ex1IntentReport rdf:type icm:IntentReport .
toco:ex1IntentReport icm:hasExpectationReport toco:
    ↪ otnDeliveryExpectationReport .
toco:otnDeliveryExpectationReport icm:target toco:
    ↪ otnScheduledBandwidthService .
toco:otnDeliveryExpectationReport icm:params toco:
    ↪ otnScheduledBandwidthParam .
toco:otnScheduledBandwidthParam icm:targetDescription toco:
    ↪ ScheduledRate_50_point0_Gbps .
toco:intentReportEndSubject toco:hasEndStatement toco:
    ↪ intentReportEndObject
```

Listing 3: Destination Graph

V. Conclusion

In this paper, we introduced GNN-ATIVE, a novel AI-native, graph-based orchestration framework designed explicitly for managing the complexities and dynamism of next-generation wireless networks. Our approach uniquely combines the strengths of GNNs and semantic KGs within a structured, hierarchical architecture (L1–L3), facilitating an intelligent, scalable, and explainable network orchestration. The proposed KG-GPT component, leveraging a customized graph-to-graph transformer architecture, has demonstrated robust reasoning capabilities by performing high-level inference directly over semantic knowledge graphs. Through experimental validation on an OTN

scenario, our framework not only showed an effective real-time responşiveness but also maintained a low inference latency which is essential for practical deployments.

Acknowledgments

The authors acknowledge the financial support by the Federal Ministry of Education and Research of Germany in the programme of "Souverän. Digital. Vernetzt." Joint project 6G-RIC, project identification numbers: 16KISK020K, 16KISK031 and 16KISK032.

References

[1] Christoph Fischer et al. "Future Integrated Mobile Communication Systems—An Outlook towards 6G". In: *International Journal of Future Computer and Communication* 11.2 (2022).

[2] *SPARQL*. https://www.w3.org/TR/sparql11-overview/. [Online; accessed 25-April-2025]. 2025.

[3] Ion Stoica et al. "A Berkeley View of Systems Challenges for Ai". In: *nil* nil.nil (2017), nil.

[4] Humza Naveed et al. "A comprehensive overview of large language models". In: *arXiv preprint arXiv:2307.06435* (2023).

[5] Heng Wang et al. "Can language models solve graph problems in natural language?" In: *Advances in Neural Information Processing Systems* 36 (2023), pp. 30840–30861.

[6] Ciyuan Peng et al. "Knowledge graphs: Opportunities and challenges". In: *Artificial Intelligence Review* 56.11 (2023), pp. 13071–13102.

[7] Mark Eisen and Alejandro Ribeiro. "Optimal Wireless Resource Allocation with Random Edge Graph Neural Networks". In: *IEEE Transactions on Signal Processing* 68 (2020), pp. 2977–2991.

[8] Yifei Shen et al. "Graph Neural Networks for Scalable Radio Resource Management: Architecture Design and Theoretical Analysis". In: *IEEE Journal on Selected Areas in Communications* 39 (1 Jan. 2021), pp. 101–115.

[9] Penghao Sun et al. "Combining Deep Reinforcement Learning with Graph Neural Networks for Optimal VNF Placement". In: *IEEE Communications Letters* 25 (1 Jan. 2021), pp. 176–180.

[10] Yanghao Xie et al. "Virtualized Network Function Forwarding Graph Placing in SDN and NFV-Enabled IoT Networks: A Graph Neural Network Assisted Deep Reinforcement Learning Method". In: *IEEE Transactions on Network and Service Management* 19 (1 Mar. 2022), pp. 524–537.

[11] Haozhe Wang et al. "A Graph Neural Network-Based Digital Twin for Network Slicing Management". In: *IEEE Transactions on Industrial Informatics* 18 (2 Feb. 2022), pp. 1367–1376.

[12] Avinash Swaminathan et al. "GraphNET: Graph Neural Networks for routing optimization in Software Defined Networks". In: *Computer Communications* 178 (Oct. 2021), pp. 169–182.

[13] Krzysztof Rusek et al. "RouteNet: Leveraging Graph Neural Networks for Network Modeling and Optimization in SDN". In: *IEEE Journal on Selected Areas in Communications* 38 (10 Oct. 2020), pp. 2260–2270.

[14] Krzysztof Rusek et al. "Unveiling the potential of Graph Neural Networks for network modeling and optimization in SDN". In: *SOSR 2019 - Proceedings of the 2019 ACM Symposium on SDN Research* (Apr. 2019), pp. 140–151.

[15] *IG1230 - Autonomous Networks Technical Architecture*. Standard. NJ, USA: TM Forum, May 2021.

[16] *IG1253 - Intent in Autonomous Networks*. Standard. NJ, USA: TM Forum, Nov. 2021.

[17] Pedro Henrique Gomes et al. "Intent-Driven Closed Loops for Autonomous Networks". In: *Journal of ICT Standardization* 9.2 (2021), pp. 257–290.

[18] ETSI. *ETSI Industry Specification Group (ISG) - Zero Touch Network and Service Management ZSM*. https://www.etsi.org/committee/zsm. [Online; accessed 25-April-2025]. 2025.

[19] W3C. *RDF Primer - Turtle version*. https://www.w3.org/2007/02/turtle/primer/. [Online; accessed 02-May-2025]. 2025.

[20] Ashish Vaswani et al. *Attention Is All You Need*. 2023. arXiv: 1706.03762 [cs.CL].

[21] Junnan Liu et al. *KnowFormer: Revisiting Transformers for Knowledge Graph Reasoning*. 2024. arXiv: 2409.12865 [cs.AI].

[22] Mihail Balanici et al. "Live demonstration of autonomous link-capacity adjustment in optical metro-aggregation networks". In: *2024 Optical Fiber Communications Conference and Exhibition (OFC)*. IEEE. 2024, pp. 1–3.

[23] Mihail Balanici et al. "Autonomous Link-Capacity Adjustment using TeraFlowSDN Controller in a Disaggregated Optical Network Testbed". In: *European Conf. on Optical Commun.(ECOC)*. 2024.

[24] Behnam Shariati et al. "Telemetry framework with data sovereignty features". In: *Optical Fiber Communication Conference*. Optica Publishing Group. 2023, M3G–2.

[25] Dave Welch et al. "Digital subcarrier multiplexing: Enabling software-configurable optical networks". In: *Journal of Lightwave Technology* 41.4 (2023), pp. 1175–1191.

[26] *IG1253A - Intent Common Model*. Standard. NJ, USA: TM Forum, Nov. 2021.

[27] Tim Dettmers et al. "Convolutional 2d Knowledge Graph Embeddings". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 32.1 (2018).