Node-Level KPI Regression over Graph-Structured Mobile Network Data: An Empirical Investigation

Guilherme F. R. Santos [1,2], **Amadeu do Nascimento Junior** [1]

[1]Ericsson Research Brazil – Indaiatuba, SP, Brazil

[2]Departamento de Computação de Sorocaba (DComp-SO)

Universidade Federal de São de Carlos (UFSCar) – Sorocaba, SP – Brazil

{guilherme.santos, amadeu.nascimento.junior}@ericsson.com

Abstract. *6G RAN management demands autonomous solutions translating high-level QoS/QoE objectives into configuration changes with measurable KPI impact. We address KPI prediction from configuration adjustments—a problem largely underexplored outside temporal settings—framed as node-level regression on graph-structured data. Through a principled ablation study, we show that lightweight GNNs outperform flat baselines by capturing topological dependencies, that KPI responses are predominantly local, and that residual connections with dedicated MLP blocks yield consistent gains, while self-supervised pretraining fails to transfer to this setting. Our results provide an empirical foundation for AI/ML-driven autonomous next-generation RAN management.*

1. Introduction

6G brings new use cases, greater demands and more complex operational scenarios for mobile networks, where artificial intelligence (AI) is offering solutions to support evolving needs. Mobile networks are now expected to support diverse services, from high-speed mobile broadband to mission-critical ultrareliable low-latency communications. As network size and service diversity grow, adopting new RAN management becomes beneficial to accommodate the new use cases envisioned for 6G, motivating a shift toward AI-native and intent-driven autonomous network management [Biyar et al. 2025]. In the AI-native and intent-based perspective envisioned for 6G [Hexa-X 2023], management is organized around closed control loops translating high-level intents into configuration actions. Intents specify desired outcomes such as Quality of Service (QoS) and Quality of Experience (QoE) targets, and at the RAN level they are expressed as requirements on Key Performance Indicators (KPIs). Closed loops observe system conditions, predict the effect of candidate actions, and decide on the best alternative to achieve the target. To realize such a scenario, autonomous operations rely on predictive models capturing how performance evolves under potential changes to configuration parameters (CPs). These models generalize system behavior and forecast KPI values to CP modifications such as antenna tilt and transmit power adjustments, serving as partial world models of RAN that allow internal simulation of future states prior to actual deployment of configurations [Sha et al. 2024].

In the literature, AI/ML models for RAN have been widely applied to cell-level or network-wide optimization tasks, such as resource allocation, interference mitigation, and

traffic forecasting [Rydén et al. 2023], typically relying on aggregated representations or predicting absolute performance metrics without explicitly modeling the effect of configuration changes. However, autonomous closed-loop control requires models that estimate the impact of candidate actions, shifting the problem toward predicting KPI variations at the granularity of individual cells conditioned on configuration parameter (CP) changes. Although graph-based models have been explored for RAN data, methodological guidelines and benchmark settings for this node-level, action-conditioned regression problem remain limited, particularly due to the challenges introduced by spatial heterogeneity, localized interference, and topology-dependent effects.

In this study, we address this gap by systematically investigating node-level CP–KPI regression through a principled ablation study. Starting from established baselines, including XGBoost and MLP, we progressively introduce graph-aware components and evaluate their individual contributions. We analyze the role of topological dependencies through GNN-based models and study the effect of increasing the receptive field via multi-hop neighborhood aggregation, providing insights into the local nature of the CP–KPI relationship. We also investigate the impact of architectural components such as residual connections and dedicated MLP blocks for parameter processing. Finally, we examine GraphMAE [Hou et al. 2022] as a self-supervised control experiment, evaluating whether its masked reconstruction pretraining objective transfers effectively to node-level continuous regression.

Contributions: Our main contributions are threefold. (i) We conduct a principled architecture search through ablation, starting from non-graph baselines and incrementally adding graph-aware components — local GNN layers, dedicated action encoding, and residual connections — to identify which design choices effectively address the task across 13 KPIs in a realistic 5G RAN simulation. (ii) We formalize node-level KPI variation prediction as a regression task over graph-structured RAN data, with an evaluation protocol and task-aligned metrics (masked Relative Error and Probability of Wrong Direction) suitable for closed-loop control. (iii) We report a negative result on GraphMAE-style self-supervised pretraining for this node-level continuous regression setting, showing that representations tuned for classification benchmarks do not readily transfer and highlighting the need for task-aligned model design in autonomous next-generation RAN.

The remainder of this paper is organized as follows. Section 3 formalizes the CP–KPI regression task over graph-structured RAN data. Section 4 describes the simulation environment and dataset construction. Section 5 details the experimental protocol, model architectures, and ablation strategy. Results are presented and discussed in Section 6, and Section 7 concludes the paper with a summary of findings and directions for future work.

2. Related Work

The problem addressed in this work sits at the intersection of three active research areas: AI/ML for RAN management, graph-based learning on network-structured data, and KPI prediction for autonomous network control. We review relevant contributions in each of these areas, highlighting the gap that motivates our study.

AI/ML for RAN management and optimization. The use of machine learning in mobile networks has been extensively studied, particularly for cell-level and network-wide tasks such as resource allocation, interference mitigation, mobility man-

agement, and traffic forecasting [Rydén et al. 2023]. These works typically operate on aggregated representations of network state, where per-cell heterogeneity is smoothed out and the prediction target is a global or sector-level quantity. In the context of 6G, AI-native and intent-driven architectures introduce a different requirement: closed-loop controllers must predict the effect of specific configuration parameter changes at the granularity of individual cells, in order to evaluate candidate actions before deployment [Hexa-X 2023, Sha et al. 2024]. This node-level, differential prediction setting, where the target is the KPI variation induced by a continuous-valued CP modification rather than an absolute performance forecast, is fundamentally distinct from prior formulations and introduces challenges that existing pipelines were not designed to address. More recently, autonomous conflict handling in intent-based management has further motivated the need for accurate node-level predictive models [Biyar et al. 2025], yet dedicated studies for this fine-grained regression task remain scarce.

Graph neural networks for network-structured data. Graph Neural Networks (GNNs) [Gilmer et al. 2017, Kipf and Welling 2017] have emerged as a natural fit for modeling communication networks, where the topology encodes meaningful relational structure between cells. In the cellular domain, spatiotemporal GNNs have been applied to traffic forecasting, fault detection, and performance prediction, capturing both spatial dependencies between neighboring cells and temporal dynamics [Lin et al. 2025]. Architectures such as GCN [Kipf and Welling 2017] and message-passing networks [Gilmer et al. 2017] form the backbone of many such approaches. Beyond standard supervised GNNs, self-supervised methods like GraphMAE [Hou et al. 2022] have demonstrated strong representation learning on graph-structured data by pretraining encoders through masked node feature reconstruction. However, these models were primarily designed and validated for node and graph classification benchmarks, with limited investigation into their applicability to node-level continuous regression, the setting relevant to our problem.

KPI prediction in RAN. Temesgene et al. [Temesgene et al. 2024] propose GNNs as prediction agents within intent-based zero-touch network management, demonstrating that graph-structured models outperform traditional neural networks in modeling network behavior. Building on this, Silva et al. [Silva et al. 2025] extend the framework to a realistic 5G RAN simulator, showing that a GNN-based prediction agent can guide the Intent Management Function (IMF) to near-optimal decisions in terms of aggregated utility. These works share our motivation of using GNNs as predictive components within closed-loop RAN management, but differ in a key aspect: both operate on graphs where nodes represent users and discrete actions, with absolute KPI values as prediction targets. However, modeling users as nodes is not realistic in real-world deployments, as fine-grained user-level data is typically not available at the management layer (e.g., SMO), where decisions are made. In contrast, our work addresses a fundamentally different formulation where nodes represent individual network cells and the regression target is the *differential* KPI response induced by a continuous-valued CP modification, a node-level delta regression setting that introduces distinct challenges, including spatial heterogeneity, localized interference patterns, and topology-dependent propagation effects not addressed by prior frameworks.

More broadly, the literature on KPI prediction in RAN has largely focused on ab-

solute performance forecasting or network-wide optimization tasks [Rydén et al. 2023], rather than on modeling the differential impact of specific configuration parameter changes at individual node granularity. This scarcity of dedicated studies for node-level CP–KPI regression limits the availability of established methodological baselines and evaluation protocols for this task, which our work directly addresses through a systematic ablation study grounded in well-known baselines.

3. Problem Formulation

We consider the problem of modeling the effect of configuration changes on RAN performance at the cell level. Given the current network state and a candidate configuration action, the objective is to predict how key performance indicators will evolve. To capture both spatial dependencies and inter-cell interference, we represent the RAN as a graph and formulate the problem as learning a transition model over graph-structured states.

3.1. Network State and Transition Model

We model the RAN as a graph $\mathcal{G}_t = (\mathcal{V}, \mathcal{E}, X_t, Z_t)$, where nodes $\mathcal{V}$ correspond to individual cells and edges $\mathcal{E}$ connect neighboring cells. The node feature matrix X_t encodes cell-level information including static descriptors, performance indicators, and configurable parameters. Edge features Z_t describe inter-cell relationships.

Configuration changes are represented as an action vector $\boldsymbol{a}_t \in \mathbb{R}^3$ which impacts cell and edge features. The network dynamics under a configuration change is modeled as:

$$\mathcal{G}_{t+1} = f_\theta(\mathcal{G}_t, \boldsymbol{a}_t) \tag{1}$$

where f_θ maps the current graph state $\mathcal{G}_t$ and action $\boldsymbol{a}_t$ to the resulting state $\mathcal{G}_{t+1}$.

This transition is illustrated in Figure 1a, where a network graph $\mathcal{G}_t$ undergoes a configuration change via action $\boldsymbol{a}_t$, producing a modified graph $\mathcal{G}_{t+1}$ with updated KPI values.

3.2. Regression Target

Rather than predicting absolute KPI values, the model targets the KPI variation induced by the configuration change:

$$\Delta K = K_{t+1} - K_t \tag{2}$$

This formulation is better aligned with the use case of closed-loop RAN management, where the controller needs to anticipate the *effect* of an action rather than the resulting absolute state. Predicting ΔK also reduces the influence of slowly-varying background conditions that are constant across the transition, focusing the model capacity on capturing the configuration-dependent response.

Concretely, the predictive model receives as input the current network graph $\mathcal{G}_t$ and the configuration action $\boldsymbol{a}_t$, and produces as output the predicted KPI delta $\widehat{\Delta K}$, as illustrated in Figure 1b.

3.3. Evaluation Metrics

Relative Error (RE). The primary evaluation metric is a masked relative error, defined as:

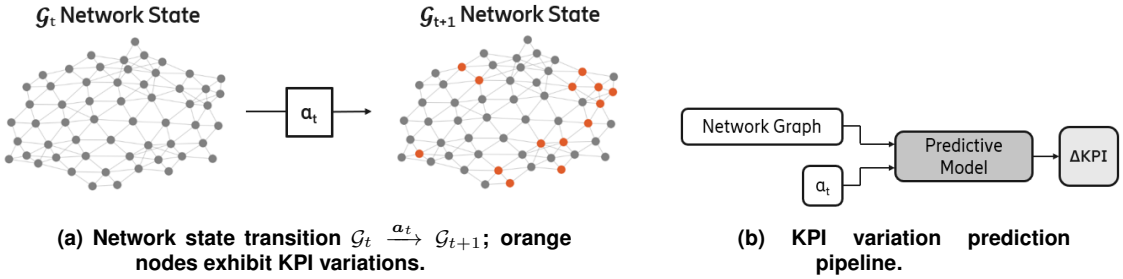
$$\text{RE} = \frac{\left|\widehat{\Delta K} - \Delta K\right|}{|\Delta K|} \tag{3}$$

Since the relative error is numerically unstable when $\Delta K \approx 0$, a threshold is considered and the metric is computed only for $\Delta K \geq 0.05$. The threshold value corresponds to the minimum KPI variation considered meaningful for closed-loop decision-making, as changes smaller than this magnitude are unlikely to influence configuration decisions. The threshold is applied uniformly across all 13 KPIs.

Probability of Wrong Direction (PWD). Beyond magnitude, a critical property of a RAN prediction model is whether it correctly identifies the *direction* of the KPI change since direction indicates whether an action improves or degrades a KPI. PWD measures the fraction of samples where the predicted and true deltas have opposite signs, restricted to the same set of measurable samples:

$$\text{PWD} = \frac{1}{m}\sum_{i=1}^{m} 1\left(\Delta K_i \cdot \widehat{\Delta K}_i < 0\right) \tag{4}$$

where m is the number of samples with $\Delta K \geq 0.05$



(a) Network state transition $\mathcal{G}_t \xrightarrow{a_t} \mathcal{G}_{t+1}$; orange nodes exhibit KPI variations.

(b) KPI variation prediction pipeline.

Figure 1. Problem formulation: network state transition and prediction pipeline.

4. Dataset

The transition model defined in Section 3 requires paired observations of the network states before and after configuration changes. Since controlled, isolated transitions of this form are not available in operational networks where multiple factors evolve simultaneously, we rely on simulation to construct a dataset of action-induced state transitions.

4.1. Simulation Environment

Data is generated using a proprietary static RAN simulator developed at Ericsson. The simulator is designed to faithfully emulate realistic multi-cell 5G NR deployments, producing aggregated cell-level KPIs that closely reflect real-world network behavior. All

exposed quantities correspond to observables available in commercial RAN nodes, ensuring that the feature space aligns with real deployment constraints rather than simulation artifacts.

The environment follows a 3GPP-compliant New Radio Urban Macro setting (TR 38.901 v16.0.0), with a hexagonal homogeneous cellular network comprising 19 sites and three sectors per site. Multiple carriers and layers are considered, leading to graphs with 57 to 171 cells depending on the active configuration. This controlled, high-fidelity setting ensures that the causal relationship between configuration changes and KPI responses is unambiguous, isolating modeling challenges from the confounds and noise inherent to operational data.

4.2. State Representation

Each simulated network state is represented as a graph $\mathcal{G}_t = (\mathcal{V}, \mathcal{E}, X_t, Z_t)$, where node features X_t are derived from three groups of observables:

- **Static descriptors (S):** deployment-related parameters such as antenna configuration, carrier frequency, bandwidth, and inter-site distance, defining the physical context of each cell.
- **Dynamic performance indicators (K_t):** aggregated cell-level measurements capturing network conditions in terms of coverage, quality, efficiency, and load. These features serve both as part of the input state and as the basis for the regression targets defined in Section 3.
- **Configurable parameters (P_t):** control variables that can be adjusted by the network, including antenna tilt (RET), maximum downlink power, and uplink power control settings. These are the parameters affected by $\boldsymbol{a}_t$.

Edge features (Z_t) encode inter-cell relationships such as relative orientation and frequency separation, which are critical to model interference coupling.

4.3. Dataset Construction

The dataset is constructed by combining a random walk over the action space with domain randomization over environment parameters. The random walk promotes locally consistent transitions, reflecting incremental configuration changes observed in practice, while domain randomization varies deployment-level parameters – including inter-site distance, carrier frequency, bandwidth, and antenna configuration – ensuring that the dataset spans a diverse range of network scenarios and topological configurations. This diversity is further reflected in the variable graph size, which ranges from 57 to 171 cells across samples.

This combination is critical to avoid overfitting to specific network instances and to promote learning of generalizable relationships between configuration changes and performance outcomes across heterogeneous deployment conditions. The final dataset contains $N = 30{,}000$ transitions:

$$\mathcal{D} = \{(\mathcal{G}_t^n, \boldsymbol{a}_t^n, \mathcal{G}_{t+1}^n)\}_{n=1}^N$$

with feature cardinalities $|S| = 11$, $|P| = 3$, $|K| = 13$, and $|Z| = 4$.

5. Experimental Setup and Model Design

Building on the dataset described in the previous section, we now define the experimental protocol and model architectures used to evaluate how different design choices affect the prediction of KPI variations. The dataset provides paired observations of network states and configuration changes, enabling a supervised learning formulation where models are trained to predict the impact of CP modifications on cell-level KPIs. In this section, we first describe the training and evaluation setup, and then introduce the sequence of models used in the ablation study.

5.1. Experimental Protocol

Only cells directly subject to CP changes are included in the supervised training signal. Cells that do not undergo parameter modifications remain in the graph as structural context for message passing but are excluded from the regression target computation.

The dataset is partitioned into fixed train and test splits shared across all models, ensuring that every approach is evaluated on identical held-out data and enabling direct, fair comparison. Concretely, given N total samples, the test set contains $\lfloor 0.2N \rfloor$ samples, while the remaining $\lfloor 0.8N \rfloor$ are used for training and validation, yielding an approximate 72/8/20 train/val/test split. The validation set is used exclusively for hyperparameter selection and early stopping.

Hyperparameters are tuned on the validation split using the procedure described in Section 5.2. Unless otherwise stated, all neural models are trained using Root Mean Squared Error (RMSE) as the loss function, optimized with Adam, with an initial learning rate of 1×10^{-4} and a decay schedule with a reduction factor of 0.1 and a minimum learning rate of 1×10^{-6}. Training is performed with a batch size of 128, and early stopping is applied based on validation loss to prevent overfitting. To account for training variability, particularly relevant given the small magnitude of performance differences between models, each configuration is trained with five independent random seeds, and we report the mean and standard deviation of RE and PWD across runs.

5.2. Search Space and Tuning Protocol

All models are tuned using Optuna [Akiba et al. 2019] with the TPE sampler over a budget of 50 trials per model family, using the validation split defined in Section 5. This ensures consistency across all approaches and enables fair comparison between architectures. We define model-specific search spaces tailored to each model family's architectural characteristics, while maintaining a comparable search budget across experiments.

For the MLP, we tune the number of hidden layers (*discrete*, 1–4), hidden dimensionality (*categorical*: 128, 256, 512), and hidden dimensionality ratio (*discrete*, 2–4). Regarding the `GeneralConv` layers, we tune the number of attention heads (*discrete*, 4–16) and aggregation function (*categorical*: add, mean, max). The action encoder is tuned over the number of hidden layers (*discrete*, 1–4), hidden dimensionality (*categorical*: 5, 10, 20, 30) and output dimensionality (*categorical*: 5, 10, 20).

For XGBoost, we tune for each individual model the number of estimators (*discrete*, 100–1000), maximum tree depth (*discrete*, 3–10), learning rate (*log-uniform*, 0.0001–0.3), subsample ratio (*continuous*, 0.5–1.0), column sampling ratio (*continuous*, 0.5–1.0), minimum child weight (*discrete*, 1–20), and gamma (*continuous*, 0.0–5.0).

5.3. Ablation Strategy

The experimental design follows an incremental ablation strategy. We start from topology-agnostic baselines and progressively introduce architectural components to isolate the contribution of each design choice. The sequence of experiments is organized as follows: (i) non-graph baselines, (ii) graph-aware modeling, (iii) GNN depth, (iv) action encoding, (v) residual connections, and (vi) a self-supervised control based on GraphMAE. Each modification is evaluated relative to a fixed reference model, allowing a direct assessment of its impact.

Baseline Models. We begin with two models that treat each node as an independent sample: XGBoost and a Multilayer Perceptron (MLP). This formulation provides a reference point for assessing the value of explicit graph modeling. The MLP takes as input the concatenation of node features X_t and the action vector $\boldsymbol{a}_t$, and produces the predicted delta vector $\Delta K \in \mathbb{R}^{13}$. As for the XGBoost, since it is a single-output model, we train 13 independent regressors, one per KPI, each taking as input the concatenation of X_t and $\boldsymbol{a}_t$. Reported metrics are averaged across all 13 models.

The MLP architecture is determined via hyperparameter search (Section 5.2), resulting in an architecture $(input \rightarrow 256 \rightarrow 512 \rightarrow 256 \rightarrow 13)$ with ReLU activations between layers.

For XGBoost, the hyperparameters are tuned independently for each KPI model (Section 5.2), with the resulting configurations reported in Table 1.

Table 1. XGBoost hyperparameters for each KPI.

	avgRsrp	avgDlSinr	badDlSinrRatio	badRsrpRatio	dlSpectralEff	dlUserThr	dlUtil	avgUlSinr	badUlSinrRatio	ulSpectralEff	ulUtil	maxUlPowerRatio	ulUserThr
n_estimators	300	300	1000	200	300	900	500	1000	400	200	600	100	500
max_depth	9	9	10	8	9	8	7	10	9	10	9	8	7
learning_rate	5.6e-2	5.6e-2	1.1e-2	1.6e-2	5.6e-2	5.6e-2	1.3e-2	6.7e-2	5.5e-2	7.0e-2	8.7e-2	1.0e-1	1.3e-2
subsample	0.89	0.89	0.81	0.89	0.89	0.72	0.61	0.77	0.59	0.87	0.63	0.67	0.60
colsample_bytree	0.70	0.70	0.76	0.78	0.70	0.91	0.80	0.80	0.90	0.99	0.90	0.72	0.91
min_child_weight	16	16	13	11	16	16	19	4	12	10	9	20	3
gamma	4.6e-3	4.6e-3	1.2e-3	2.6e-2	4.6e-3	7.5e-2	1.2e-3	1.2e-3	9.3e-3	1.8e-1	7.6e-3	4.5e-2	2.6e-2

Graph-Aware Architecture. We extend the baseline formulation by explicitly modeling the graph structure of the RAN. We adopt the `GeneralConv` layer [Fey and Lenssen 2019] because it supports edge features and attention-based aggregation, both relevant for capturing inter-cell relationships such as inter-frequency offset and relative azimuth. To isolate the contribution of neighborhood aggregation, a single `GeneralConv` layer is prepended to the MLP head, projecting node features into an N-dimensional embedding where N matches the hidden dimensionality of the MLP head. By keeping the MLP head architecture unchanged, any performance difference can be attributed specifically to the graph convolution step and its ability to aggregate neighborhood information.

The specific configuration of the `GeneralConv` layer is selected via hyperparameter search, with the search space defined in Section 5.2. The resulting configuration employs 4 attention heads and additive aggregation.

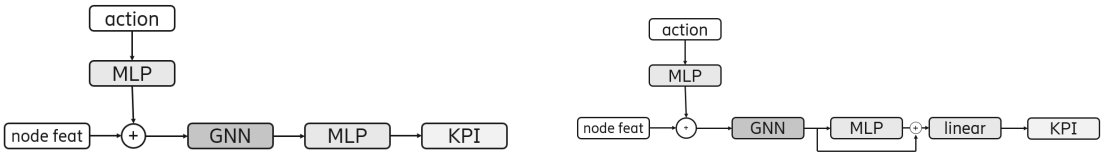
To analyze the impact of the receptive field, we also evaluate configurations with 2, 3, and 4 stacked GNN layers. In all cases, the first layer projects the input to an N-dimension embedding, and subsequent layers operate within the same embedding space.

Action encoding. We next investigate whether the representation of configuration

changes affects prediction performance. In the baseline graph formulation, the action vector $\boldsymbol{a}_t$ is concatenated directly to node features. To provide a richer representation, we introduce a dedicated action encoder. For each configuration parameter, the current value and its increment are stacked into a 2-dimensional input and processed by a shared MLP, producing a learned per-parameter embedding. This allows the model to capture the interaction between the current parameter value and the applied change. The embeddings for all $|P| = 3$ parameters are flattened, concatenated with node features, and passed to the GNN block (Figure 2a).

The action encoder architecture is determined via hyperparameter search (Section 5.2), resulting in a 4-layer MLP with dimensions $(2 \to 5 \to 20 \to 10)$ and ReLU activations between layers.

Residual connections. Finally, we evaluate residual learning [He et al. 2016] in the prediction head (Figure 2b). Residual connections are known to improve optimization in deep networks by facilitating gradient flow and enabling layers to learn corrections over their inputs rather than full transformations. In our setting, this is particularly relevant as the prediction head processes a learned representation from the GNN, and the target ΔK often consists of small variations. Residual connections allow the model to refine this intermediate representation, instead of reconstructing the output from scratch. As a control experiment, we also introduce residual connections between GNN layers to assess whether performance saturation at larger depths may be related to over-smoothing.



(a) Modified architecture with action encoder.

(b) Final architecture with residual connection.

Figure 2. Evolution of the proposed architecture.

5.4. Self-supervised Control (GraphMAE)

To evaluate whether representation learning can improve performance beyond architectural refinement alone, we include GraphMAE [Hou et al. 2022] as a self-supervised control. GraphMAE is a masked graph autoencoder that learns node representations by reconstructing masked node features.

We pretrain the GraphMAE encoder on 25,000 samples from the training partition. The model is then fine-tuned end-to-end on the downstream regression task using a two-speed learning-rate schedule: the pretrained encoder is updated at 1×10^{-5}, while the action encoder and regression head are trained at 1×10^{-4}. The downstream model combines the pretrained node embeddings with the action encoding and passes them through a regression head with a residual connection. This downstream architecture is modeled after the final proposed architecture to ensure a fair comparison. We refer to this configuration as **GraphMAE-FT**.

6. Results

We now present the results of the ablation study introduced in Section 5, following the same progression of models to quantify the contribution of each architectural component to KPI variation prediction.

Table 2 summarizes the performance of all evaluated methods and reveals a clear progression across the ablation sequence: starting from topology-agnostic baselines, each successive modification improves predictive performance, culminating in the final model, which achieves the best results across all aggregate metrics.

Table 2. Evaluated models performance over five random seeds.

Method	RE (%)	PWD (%)	RMSE	Params
XGBoost	66.28 ± 0.02	17.42 ± 0.01	0.0422 ± 0.00004	2M
MLP	57.42 ± 0.08	14.92 ± 0.09	0.0412 ± 0.00033	**274K**
1-Layer GNN	52.59 ± 0.22	10.68 ± 0.12	0.0394 ± 0.00014	314K
2-Layer GNN	52.00 ± 0.11	10.24 ± 0.01	0.0390 ± 0.00007	580K
Action Encoding	49.28 ± 0.30	9.22 ± 0.13	0.0373 ± 0.00009	625K
Final Model	**48.01** ± 0.14	**8.93** ± 0.09	**0.0369** ± 0.00007	625K
GraphMAE-FT	56.46 ± 0.15	13.81 ± 0.02	0.0404 ± 0.00004	1.3M

6.1. From Topology-Agnostic to Graph-Aware Models

We begin comparing the two topology-agnostic baselines. XGBoost achieves a RE of 66.28%, while MLP achieves a RE of 57.42%. This result indicates that a shared multi-output neural model is better suited than independent tree-based regressors to capture the non-linear relationships between CP changes and KPI responses.

These values must be interpreted in the context of the delta regression. Since many target variations are small in magnitude, even small absolute prediction errors can yield inflated relative errors for samples where ΔK is close to 0.05, as discussed in Section 3.3. Nevertheless, the comparison between XGBoost and MLP provides a consistent topology-agnostic reference for the remainder of the study.

Introducing a single graph layer yields a consistent improvement over the topology-agnostic MLP baseline, reducing RE from 57.42% to 52.59%. This gain confirms that explicit neighborhood aggregation captures inter-cell dependencies that are informative for KPI variation prediction and inaccessible to models that treat each cell independently.

To assess whether larger receptive fields provide additional gains, we evaluate GNN depths from 1 to 4 layers (Table 3). Moving from 1 to 2 layers yields a consistent improvement of 0.5 percentage points, while deeper configurations produce only marginal changes within the variance across seeds. This suggests the task is predominantly local: KPI responses are largely governed by direct neighbors, and aggregating from more distant cells offers diminishing returns. We therefore adopted the 2-layer configuration for subsequent experiments.

6.2. Architectural refinements and Representation Learning

Building on the graph-aware backbone, we next evaluate architectural refinements aimed at improving how configuration changes are represented and how predictions are

Table 3. Comparison of GNN layers on the performance over five random seeds.

Number of Layers	RE (%)	PWD (%)	RMSE	Params
1	52.59 ± 0.22	10.68 ± 0.12	0.0394 ± 0.00014	**314K**
2	$\mathbf{52.00} \pm 0.12$	10.24 ± 0.02	$\mathbf{0.0390} \pm 0.00007$	580K
3	52.05 ± 0.15	10.18 ± 0.08	0.0391 ± 0.00019	846K
4	52.07 ± 0.14	$\mathbf{10.12} \pm 0.03$	0.0395 ± 0.00004	1.1M

formed.

Adding a dedicated action encoder improves the 2-layer GNN from 52.00% to 49.28% in RE and from 10.24% to 9.22% in PWD. Although the gain is smaller than the one obtained by introducing graph structure, it is consistent across seeds and achieved with limited additional complexity. This result suggests that explicitly modeling the interaction between the current parameter value and the applied increment is beneficial. Instead of forcing the GNN to disentangle raw action values from the node state, the model receives a more structured representation of the intervention, which appears to facilitate the prediction of the resulting KPI variations.

Residual connections in the prediction head further improve performance, reducing RE from 49.28% to 48.01% and PWD from 9.22% to 8.93%. The gain is modest but consistent, reinforcing the trend observed throughout the ablation.

As a control experiment, we also introduce residual connections between GNN layers. In this case, no improvement was observed; in fact, performance degraded slightly across all evaluated depths. This result suggests that the saturation observed beyond 2 GNN layers is unlikely to be caused by over-smoothing. A more plausible explanation is that CP–KPI relationship is predominantly local, so additional propagation depth does not add useful predictive information.

Finally, we evaluate whether representation learning can provide additional gains through GraphMAE-FT pretraining. The GraphMAE-FT configuration underperforms both the final supervised model and the simple 1-layer GNN, achieving a RE of 56.46% and a PWD of 13.81%. This places it much closer to the non-graph MLP baseline than to the best graph-aware supervised models.

A likely explanation is the misalignment between the pretraining objective and the downstream task. GraphMAE is designed to reconstruct masked node features, an objective that is effective in many node and classification benchmarks. However, our task is node-level continuous regression, where the relevant signal lies in fine-grained numerical relationships between configuration changes and KPI responses. Representations optimized for mask reconstruction do not appear to transfer effectively to this setting, even after end-to-end fine-tuning.

To further investigate this point, we analyze the latent space geometry of the final supervised model and GraphMAE-FT. Specifically, we train a k-Nearest Neighbor (k-NN) regressor on the representations fed into the regression head of each model and use it to predict mean $|\Delta K|$. Because the k-NN model has no trainable parameters, its R^2 score reflects the extent to which local geometry in latent space is aligned with the downstream target. Across $k \in \{5, 10, 15, 20\}$, the supervised GNN consistently achieves $R^2 \approx 0.54$,

whereas GraphMAE-FT reaches only $R^2 \approx 0.26$. This result indicates that end-to-end supervised training produces a latent space more aligned with the regression objective.

6.3. Aggregate and Per-KPI analysis

Taken together, the aggregate results show a consistent trajectory across the ablation. Relative to the MLP baseline, the final model reduces RE by **16.38%** and PWD by **40.14%**. Relative to XGBoost, the reduction in RE exceeds **27%**. The largest improvement comes from incorporating graph structure, while action encoding and residual connections provide smaller but reliable gains that compound into the final performance advantage.

To assess whether these improvements are uniform across targets, we analyze RE on a per-KPI basis, as shown in Figure 3a. The final model achieves the lowest RE on all 13 KPIs, indicating that architectural refinements do not improve one subset of KPIs at the expense of another. This is an important property for autonomous RAN management, where consistent performance across indicators is more valuable than isolated gains on individual targets.

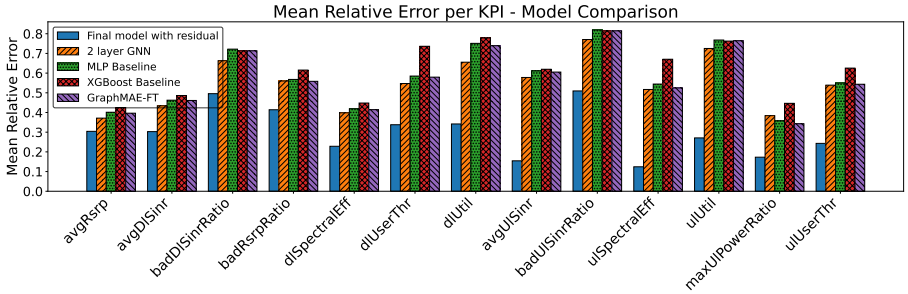
The per-KPI breakdown also reveals which indicators are inherently more challenging. Ratio-based indicators (badDlSinrRatio, badRsrpRatio, badUlSinrRatio) and utilization metrics (dlUserThr, dlUtil, ulUtil) exhibit consistently higher RE across all models, likely due to threshold effects and load-dependent dynamics. By contrast, average signal quality metrics such as avgRsrp and avgDlSinr tend to be more predictable, reflecting a more direct coupling between configuration changes and performance response.

We also examine the distribution of RE values per KPI using the violin plots in Figure 3b. Beyond improving the mean, the final model reduces the occurrence of extreme errors. This effect is particularly visible for badUlSinrRatio, where the MLP baseline exhibits large outliers, while the final model produces a substantially more conservative distribution. This suggests improved robustness on more challenging samples.

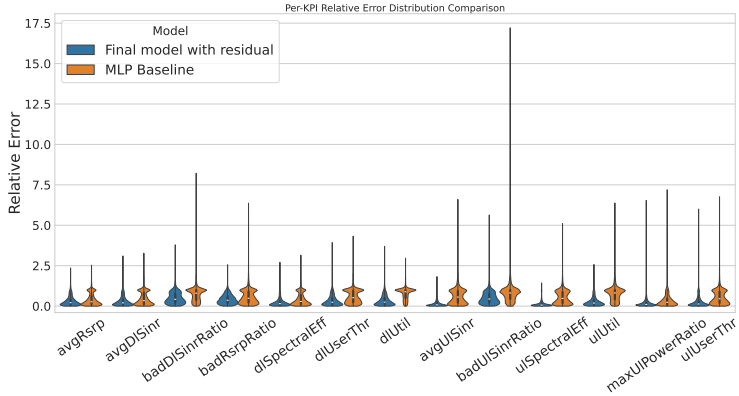
7. Conclusion

This work investigated node-level CP–KPI regression for autonomous RAN management as a prediction task over graph-structured network states. Starting from topology-agnostic baselines, we performed a controlled ablation study to identify which modeling choices are most relevant for predicting KPI variations caused by configuration changes.

The results support three main conclusions. First, explicit graph modeling is a key factor for performance: replacing a flat MLP with a GNN consistently improves both Relative Error (RE) and Probability of Wrong Direction (PWD), confirming that inter-cell dependencies contain predictive information inaccessible to topology-agnostic models. Second, the benefit of graph modeling is primarily local, increasing the receptive field beyond two GNN layers yields no additional gain, indicating that KPI responses are largely governed by immediate neighborhoods. Third, architectural refinements beyond the graph backbone provide smaller but consistent improvements: dedicated action encoding and residual connections together reduce RE by 16.38% and PWD by 40.14% relative to the MLP baseline. Consequently, graph structure is the primary driver of performance



(a) Mean Relative Error per KPI.



(b) Relative Error distribution per KPI.

Figure 3. Per-KPI evaluation of prediction errors.

gains, and further improvements depend on task-aligned modeling choices rather than generic representation learning.

In addition, we state that the self-supervised control experiment with GraphMAE does not improve the downstream regression task. In our setting, reconstruction-based pretraining underperforms even a shallow supervised GNN, suggesting that reconstruction-based pretraining objectives, which ignore action-conditioning, are not well aligned with the task.

These findings should be interpreted in light of the experimental setting. All results are obtained in a static, simulator-based environment, which provides controlled and 3GPP-compliant data but may not fully capture dynamics and distribution shifts observed in operational networks. In particular, the conclusion that local neighborhoods are sufficient may not hold in the presence of temporal dynamics or delayed inter-cell effects. At the same time, the model relies exclusively on observables available in commercial RAN nodes. While this constrains the available information, it strengthens the practical relevance of the study by aligning the feature space with real-world deployment conditions.

Future work should therefore assess whether these findings hold under temporal dynamics, more heterogeneous datasets (including but not limited to commercial deployments), and richer representations of coupled configuration changes.

8. Acknowledgments

Portions of the text were refined with generative AI tools (Microsoft Copilot with GPT 5.3 and Claude Sonnet 4.6) to improve clarity and writing quality. All technical

content, analyses, and conclusions were authored and validated by the authors.

References

Akiba, T., Sano, S., Yanase, T., Ohta, T., and Koyama, M. (2019). Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*.

Biyar, E. D. et al. (2025). Autonomous conflict handling in intent-based management. *Computer Networks*, 271:111561.

Fey, M. and Lenssen, J. E. (2019). Fast graph representation learning with pytorch geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. (2017). Neural message passing for quantum chemistry. In *Advances in Neural Information Processing Systems*, volume 30, pages 1263–1272. Curran Associates, Inc.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Hexa-X (2023). Deliverable d3.2 - initial architecture enablers. Technical report, Hexa-X.

Hou, Z., Liu, X., Cen, Y., Dong, Y., Yang, H., Wang, C., and Tang, J. (2022). Graphmae: Self-supervised masked graph autoencoders. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.

Kipf, T. N. and Welling, M. (2017). Semi-supervised classification with graph convolutional networks. In *Proceedings of the 5th International Conference on Learning Representations (ICLR)*.

Lin, J., Lan, T., Zhang, B., Lin, K., Miao, D., He, H., Ye, J., Zhang, C., and Li, Y.-F. (2025). Multi-scenario cellular kpi prediction based on spatiotemporal graph neural network. *IEEE Transactions on Automation Science and Engineering*, 22:5131–5142.

Rydén, H. et al. (2023). Next generation mobile networks' enablers: Machine learning-assisted mobility, traffic, and radio channel prediction. *IEEE Communications Magazine*, 61(10):94–98.

Sha, T., Zhang, Y., Li, Q., Zhang, Z., Zhu, L., Hua, X., Yu, R., Fan, X., Lei, Z., Feng, J., and Zhang, Y. (2024). World model aided parameter adjustment decision and evaluation system for radio access network. In *ICC 2024 - IEEE International Conference on Communications*, pages 1649–1654.

Silva, A., Temesgene, D. A., Klautau, A., Aben-Athar, R., and Nahum, C. (2025). Leveraging GNNs for intent-driven 5G RAN optimization in autonomous networks. *IEEE Access*, 13:189096–189110.

Temesgene, D. A., Biyar, E. D., Silva, A., Likhyani, A., and Zahemszky, A. (2024). Graph neural network for building prediction agents in intent-based zero-touch networks. In *Proceedings of the IEEE International Conference on Communications (ICC)*, pages 974–979.