Multi-Task Learning Architectures for Joint Interference Detection and KPI Prediction in 5G Networks

Mina Kaviani[1], Jurandy Almeida[1], Fábio L. Verdi[1], Ricardo Souza[2]

[1] Department of Computer Science (DComp) – Federal University of São Carlos (UFSCar) Sorocaba – SP – Brazil

[2]Ericsson Research Indaiatuba – SP – Brazil

mina.kaviani@estudante.ufscar.br, {verdi,jurandy.almeida}@ufscar.br,

ricardo.s.souza@ericsson.com

Abstract. *Real-time interference detection and accurate Key Performance Indicator (KPI) prediction are critical for optimizing 5G radio access networks. Jointly addressing these objectives offers a comprehensive view of network behavior but presents a significant challenge: simultaneously optimizing for heterogeneous tasks—discrete interference classification and continuous KPI regression—often leads to negative transfer or inefficient parameterization. In this paper, we systematically investigate the effectiveness of Multi-Task Learning (MTL) for this dual objective by evaluating distinct architectural strategies, including Hard Parameter Sharing, Cross-Stitch networks, Multi-gate Mixture-of-Experts (MMoE), Progressive Layered Extraction (PLE), and an attention-based model, against a Single-Task Learning (STL) baseline. Using a dataset collected from a realistic 5G testbed, we analyze the trade-offs between classification accuracy, regression error, model complexity, and inference latency. Our experimental results demonstrate that no single architecture dominates across all metrics. While STL achieves high predictive performance, it is computationally prohibitive for real-time applications due to redundant feature extraction. Conversely, Hard Parameter Sharing offers minimal latency but suffers severe performance degradation due to rigid representation sharing. PLE delivers the highest classification accuracy (87.62%) but at the cost of increased model size. Ultimately, MMoE emerges as the optimal architecture for practical deployment; it achieves the lowest total test loss and high classification accuracy (86.80%) while reducing Floating Point Operations (FLOPs) by approximately 74% compared to STL, making it well-suited for practical interference-aware monitoring and optimization in 5G radio access networks.*

1. Introduction

With the rapid deployment of 5G networks and the increasing demand for high-reliability low-latency communication, real-time interference detection and accurate performance monitoring have become critical requirements for modern Radio Access Networks (RAN). Interference—whether from internal co-channel sources or external jammers—can significantly degrade network capacity, user throughput, and overall system reliability. To maintain Quality of Service (QoS), network operators must not only detect the

presence of interference immediately but also continuously monitor Key Performance Indicators (KPIs) such as Reference Signal Received Power (RSRP), Reference Signal Received Quality (RSRQ), and Signal-to-Interference-plus-Noise Ratio (SINR).

Jointly addressing interference detection and KPI prediction offers a comprehensive view of the radio environment, enabling proactive optimization and automated root-cause analysis. However, simultaneously modeling these tasks is non-trivial due to their heterogeneous nature: interference detection is a discrete classification problem requiring sharp decision boundaries, whereas KPI prediction involves continuous-valued regression with distinct sensitivity and noise characteristics. Naïvely optimizing these conflicting objectives in isolation—or via simplistic shared representations—often leads to negative transfer, where the dominant task suppresses the learning of the auxiliary task, or to inefficient over-parameterization that exceeds the tight computational budgets of edge-deployed 5G network devices.

Multi-Task Learning (MTL) has emerged as a promising paradigm to address these challenges by exploiting shared representations across related tasks. While MTL has been applied to specific 5G domains such as traffic forecasting [Tommy et al. 2025] or signal characterization [Jagannath and Jagannath 2022], existing literature largely focuses on homogeneous tasks (e.g., regression-only or classification-only) or relies on single architectural designs without exploring the spectrum of feature-sharing strategies. There is a lack of systematic comparisons that specifically address the trade-offs between regression fidelity, classification accuracy, and computational efficiency in the context of hybrid radio tasks.

The objective of this paper is to fill this gap by systematically investigating the effectiveness of MTL architectures for jointly addressing interference classification and KPI regression. We conduct a comprehensive evaluation of representative strategies, ranging from rigid Hard Parameter Sharing to flexible, expert-based architectures including Cross-Stitch Networks, Multi-gate Mixture-of-Experts (MMoE), Progressive Layered Extraction (PLE), and Attention-based models. We benchmark these against a Single-Task Learning (STL) baseline using a real-world 5G dataset capturing complex interference scenarios on an experimental testbed.

The contributions of this paper are twofold. First, we provide a comparative study of diverse MTL mechanisms—ranging from implicit regularization in Hard Parameter Sharing to explicit task routing in Mixture-of-Experts—applied specifically to the heterogeneous problem of interference detection and KPI regression. Second, we move beyond standard accuracy metrics to evaluate the practical feasibility of these models for 5G deployment. By correlating predictive performance with inference latency and computational complexity (FLOPs), we establish a decision framework that helps network architects balance the conflicting requirements of high-fidelity monitoring and real-time processing in intelligent 5G network applications.

This paper is organized as follows. Section 2 reviews related work on multi-task learning in 5G networks. Section 3 describes the problem formulation and the evaluated MTL architectures. Section 4 presents the experimental setup, results, and a detailed trade-off analysis. Finally, Section 5 concludes the paper and outlines future research directions.

2. Related works

The application of MTL in 5G network optimization has gained significant traction, primarily driven by the need to efficiently manage scarce radio resources and computational power. For instance, [Tommy et al. 2025] proposed a novel MTL framework for spatio-temporal beam-level traffic forecasting in 5G networks. The approach leverages shared representations across multiple prediction tasks, effectively addressing challenges such as intermittent data patterns, limited time-series lengths, and multivariate complexity. Experiments using a high-resolution, beam-level dataset from real-world 5G deployments demonstrate that the MTL framework significantly outperforms conventional models like Long Short-Term Memory (LSTM), achieving much lower prediction errors (MSE of 0.00428 and MAPE of 0.4% versus 0.079 and 36.5% for LSTM). These results highlight the potential of applying MTL for robust and proactive network optimization in dynamic 5G environments.

Similarly, [Jagannath and Jagannath 2022] addressed the challenge of spectrum awareness in future communication networks by proposing a deep learning–based MTL framework that jointly performs modulation and signal classification over heterogeneous radar and communication waveforms. Their lightweight architecture exploits task correlation to improve classification accuracy and learning efficiency while supporting edge deployment. Experimental evaluations using over-the-air data show notable gains in accuracy, computational complexity, and memory efficiency compared to reference models.

Expanding on this, [Wang et al. 2024] addressed wideband signal recognition in cognitive wireless communications by employing an MTL framework that jointly performs spectrum sensing, modulation recognition, and signal classification. By utilizing shared feature extraction and joint optimization, their approach effectively exploits task correlations, leading to improved recognition accuracy and robustness compared to conventional single-task methods, particularly in complex and dynamic wideband signal environments.

While these works demonstrate the efficacy of MTL in 5G, they predominantly address homogeneous tasks—either purely regression (traffic) or purely classification (modulation). They rarely address the complexities of jointly modeling heterogeneous objectives, where the risk of negative transfer is significantly higher.

In the broader machine learning community, many works have focused on advancing the MTL. The study done in [Zhang et al. 2022] demonstrates that, contrary to conventional hard parameter sharing in multi-domain learning, using domain-specific parameters in the lower layers can significantly improve performance. It also demonstrates that models with only a small amount of domain-specific bottom-layer parameters can achieve performance comparable to fully independent models, suggesting a stronger baseline for multi-domain learning design.

As discussed in [Misra et al. 2016], MTL in convolutional neural networks can effectively improve recognition performance by learning shared representations across tasks. Moreover, introducing cross-stitch units enables adaptive, end-to-end learning of an optimal balance between shared and task-specific features, leading to better generalization, particularly in scenarios with limited training data.

The study presented in [Ma et al. 2018] highlights that neural-based MTL is

widely used in large-scale applications such as recommendation systems. However, its performance is sensitive to task-relatedness. To address this limitation, the MMoE framework learns task relationships through shared experts and task-specific gating networks, enabling flexible knowledge sharing and improved performance across diverse tasks.

Finally, the work [Tang et al. 2020] investigates MTL in recommendation systems and highlights its limitations caused by negative transfer arising from complex and competing task correlations. The authors further observe a seesaw phenomenon, where performance improvements in one task may degrade others. To address these challenges, the proposed PLE model separates shared and task-specific components and progressively learns deeper semantic representations. Experimental results on large-scale real-world datasets demonstrate that PLE outperforms state-of-the-art MTL approaches in both offline and online recommendation scenarios.

Despite these innovations, their systematic evaluation in the context of 5G networks remains limited. As summarized in Table 1, most 5G-centric studies adopt a single MTL architecture without exploring the trade-offs between model complexity, inference latency, and task balance. In contrast, this work focuses on the joint modeling of interference detection (classification) and KPI regression—a heterogeneous problem prone to negative transfer. We systematically compare representative architectures (Hard Sharing, Cross-Stitch, MMoE, PLE, and Attention-based models) to identify the optimal strategy for balancing predictive accuracy and computational efficiency in real-time 5G monitoring.

Table 1. Comparison of related works on MTL in 5g networks

Work	**Learning Objective**
Tommy [Tommy et al. 2025]	Regression-based MTL for Traffic Forecasting
Jagannath [Jagannath and Jagannath 2022]	Homogeneous MTL for Signal & Modulation Classification
Wang [Wang et al. 2024]	Classification-based MTL for Signal Recognition
Misra [Misra et al. 2016]	Cross-Stitch Networks for MTL
Zhang [Zhang et al. 2022]	Architecture-level Analysis of MTL Parameter Sharing
Ma [Ma et al. 2018]	MMoE Architecture for MTL
Tang [Tang et al. 2020]	PLE-based MTL for Recommendation Systems
This work	Comparative Hybrid MTL (Cls+Reg) for Interference-Aware KPI Prediction

3. System model

In this paper, we investigate the effectiveness of MTL compared to STL for the joint problem of interference detection and KPI prediction in 5G networks. The proposed framework exploits temporal radio measurements to jointly perform interference classification and KPI regression, enabling a systematic comparison of different learning architectures.

3.1. Deep Learning–Based Single-Task and Multi-Task Learning Models

Building on prior studies in 5G performance prediction, we employ LSTM-based deep learning models due to their strong capability in capturing temporal dependencies in sequential radio measurements. In this work, both STL and MTL architectures are considered, including Attention-based model, hard parameter sharing, Cross-Stitch networks,

PLE and MMoE. These architectures are well suited for jointly addressing interference detection and KPI prediction tasks. Rather than focusing on architectural novelty, our objective is to provide a fair and systematic comparison of different learning strategies under a unified temporal modeling framework.

Figure 1 illustrates the STL model used as a baseline, in which each prediction task is learned independently using a dedicated LSTM encoder and a task-specific prediction head. Separate LSTM networks are employed for RSRP, RSRQ, SINR regression, and interference classification, with no parameter sharing across tasks. Each LSTM extracts task-specific temporal representations from the same input sequence, which are mapped to the corresponding outputs through fully connected layers, providing a clear reference for evaluating the benefits of parameter sharing in MTL architectures.

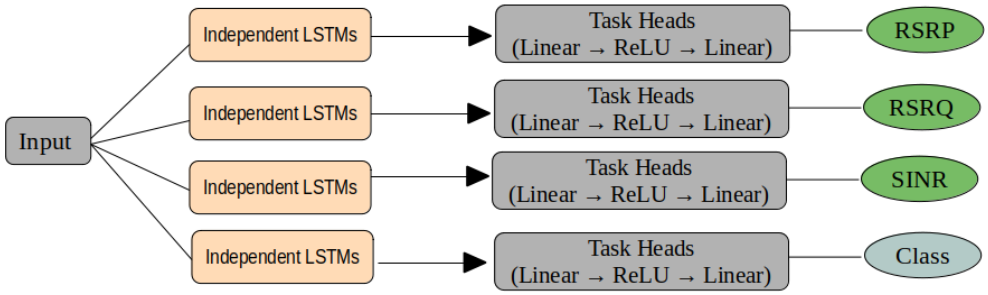

Figure 1. Architecture of the STL model.

We adopt a Cross-Stitch Network to jointly model KPI regression (RSRP, RSRQ, SINR) and interference classification. Each task has a separate LSTM-based feature extractor, and task representations are combined via Cross-Stitch Units, which learn a linear combination of features across tasks:

$$\tilde{\mathbf{x}}^{(i)} = \sum_{j=1}^{T} \alpha_{ij} \mathbf{x}^{(j)}, \quad i = 1, \ldots, T \tag{1}$$

where T is the number of tasks and α_{ij} are learnable parameters controlling cross-task sharing. Values near zero preserve task-specific features, while higher values encourage shared representations. The Cross-Stitch parameters are initialized as an identity matrix, allowing the model to gradually learn beneficial sharing in an end-to-end manner.

As illustrated in Figure 2, multiple Cross-Stitch Units are stacked between task-specific LSTM encoders and output heads. This design adaptively balances shared and task-specific features, improving robustness to negative transfer while enabling efficient joint learning of regression and classification objectives [Misra et al. 2016].

PLE is a MTL model designed to handle complex and potentially conflicting task correlations. Unlike traditional MTL models, PLE explicitly separates shared and task-specific experts to reduce negative transfer. It employs a progressive routing mechanism with multi-level experts and gating networks to gradually extract deeper semantic knowledge from lower layers while separating task-specific parameters in higher layers. This

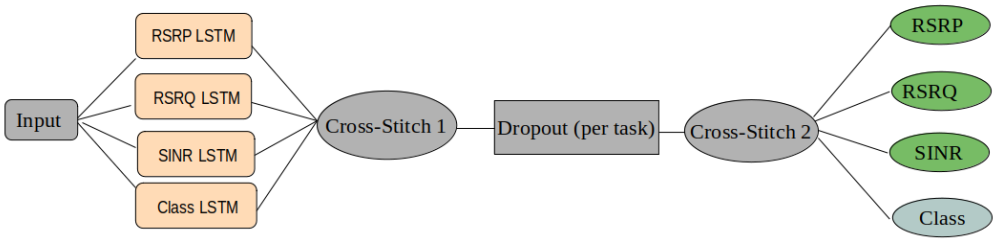

Figure 2. Architecture of the Cross-Stitch Network for MTL.

structure improves the efficiency of joint representation learning and information routing across tasks, effectively mitigating the seesaw phenomenon, where improving one task may degrade the performance of others. PLE has been shown to outperform state-of-the-art MTL models in both industrial and public benchmark datasets [Tang et al. 2020]. The overall architecture is illustrated in Figure 3.

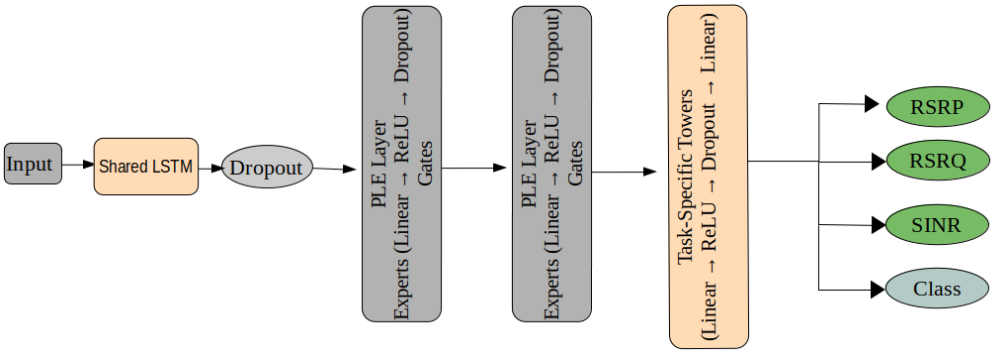

Figure 3. Architecture of the PLE model for MTL.

MMoE is a neural MTL architecture that models task relationships by sharing a set of expert subnetworks across all tasks while employing task-specific gating networks to dynamically combine these experts. Each task learns its own gating function, which assigns different importance weights to the shared experts, enabling flexible parameter sharing and effective handling of varying task relatedness [Ma et al. 2018]. Figure 4 illustrates the architecture of the MMoE model used in this work.

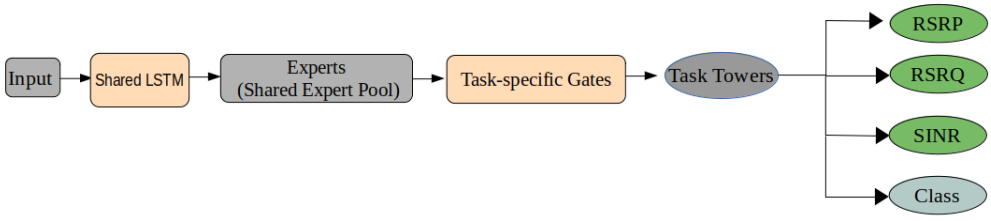

Figure 4. Architecture of the MMOE model for MTL.

A task-specific Attention mechanism is applied over the shared LSTM outputs, where each task uses a learnable query vector to compute scaled dot-product Attention weights across time steps [Vaswani et al. 2017]. This design (Figure 5) is adapted to MTL

by enabling task-aware temporal feature aggregation without relying on a Transformer architecture.

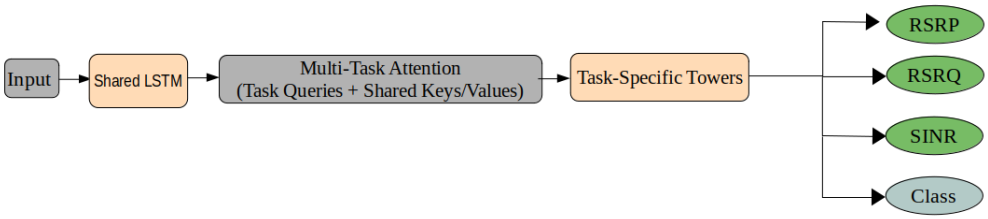

Figure 5. Architecture of the Attention-based MTL model.

Hard parameter sharing is a classical MTL strategy in which all tasks share a common feature extractor, while task-specific output towers are learned independently. In this model, a shared LSTM encodes the input sequence into a single latent representation that is directly fed to all regression and classification heads, enforcing full parameter sharing at the representation level (Figure 6). This approach reduces model complexity and acts as an implicit regularizer, but may suffer from negative transfer when task objectives are weakly correlated [Caruana 1997].

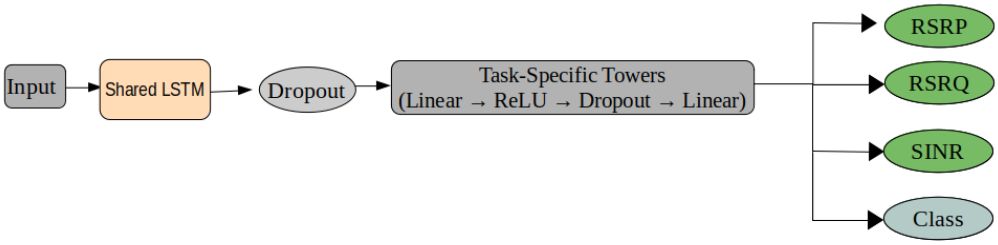

Figure 6. Architecture of the Hard parameter sharing model for MTL.

4. Experimental evaluation

This section reveals the intricate details of our data collection and methodology, meticulously outlining each step and ensuring a clear understanding of the research framework.

4.1. Data collection and preparation

The FccIQ Dataset [FCCLab 2025] is collected from a live 5G NR experimental testbed built on the NVIDIA Aerial CUDA-accelerated RAN, providing a realistic representation of radio behavior under diverse interference conditions. The dataset jointly includes Layer-1 baseband I/Q samples and Layer-2/3 radio and transport KPIs captured at a 20 Hz sampling rate (50 ms) using an xApp-based monitoring architecture integrated with the RAN Intelligent Controller (RIC). Layer-1 I/Q samples are autonomously collected by the CUDA-accelerated cuBB and stored in a high-performance ClickHouse database, while Layer-2/3 metrics are gathered via a dedicated xApp monitoring script. The recorded features span the PHY layer (e.g., RSRP, SS-RSRP, SS-RSRQ, SS-SINR, CQI), MAC (BLER, HARQ statistics, MCS, PHR), RLC (buffer status and retransmissions), PDCP (throughput and sequence tracking), and GTP (tunnel and packet statistics).

Data is captured under controlled and realistic interference scenarios, including UE-to-BS co-channel interference, BS-to-BS TDD mismatch, and external jamming, with uplink throughput benchmarked using iPerf3 under both high- and low-load configurations. Overall, FccIQ Dataset 01 enables comprehensive evaluation of ML-based throughput estimation, adaptive AI model partitioning between UE and edge, and performance analysis of 5G NR systems operating in dynamic and interference-prone environments.

We address the issue of missing values by representing unavailable entries using NaN placeholders, which are natively supported in Python-based numerical computations. Data normalization is performed using the MinMaxScaler to ensure consistent feature ranges and facilitate more stable training. To maintain temporal dependencies in sequential data, we adopt a sliding window strategy with a window length of $w = 36$, shifted T times along the sequence. This process preserves the temporal structure of the data and is illustrated in Figure 7 [Parera et al. 2019].

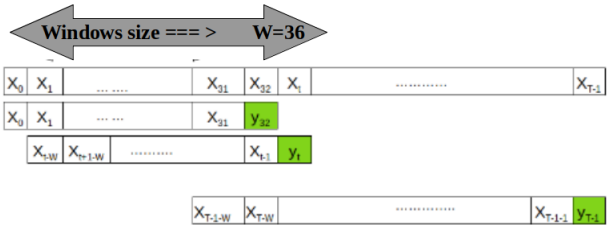

Figure 7. Sliding windows for time series forecasting.

4.2. Training procedure

To leverage the inherent time dependence of our data, we strategically partition it by timestamps, reflecting real-world dynamics where past information heavily influences future predictions. Each CSV dataset is meticulously divided, dedicating 80% for training and reserving 20% for rigorous testing.

4.3. Implementation details

We implemented the Python code for our project and executed it using Google Colab, a cloud-based platform for collaborative coding and data analysis. We also defined the network architecture and training hyperparameters for the models, which are summarized in Tables 2 and 3, while the configurations of the other evaluated models are available on our GitHub repository[1].

4.4. Performance metrics

The proposed MTL models are evaluated on both regression and classification tasks. For the regression tasks, which aim to predict key radio signal quality indicators including RSRP, RSRQ, and SINR, the Mean Squared Error (MSE) is used as the primary evaluation metric. The MSE for each regression task is defined as [Goodfellow et al. 2016]:

$$\mathcal{L}_{\text{MSE}} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2, \tag{2}$$

[1] https://github.com/dcomp-leris/FccIQ_Dataset_MTL.

Table 2. STL-LSTM Model Configuration: (a) Network Architecture and (b) Training Hyperparameters

(a) Network Architecture

Component	Configuration
Input Size	4
Window Size	36
Hidden Size	64
Number of LSTM Layers	1
Dropout	0.3
FC Layers per Task	1
FC Hidden Size	64
FC Activation	ReLU
Regression Output Size	1
Total Regression Outputs	3

(b) Training Hyperparameters

Parameter	Value
Optimizer	Adam
Initial Learning Rate	0.001
Number of Epochs	300
Mini-Batch Size	24

Table 3. MMoE LSTM Configuration: (a) Network Architecture and (b) Training Settings

(a) Network Architecture

Component	Configuration
Input Size	4
Window Size	36
Shared LSTM Hidden Size	64
Number of LSTM Layers	1
Number of Experts	2
Expert Hidden Size	64
Number of Gates	4
FC Layers per Task	2
FC Activation	ReLU
Regression Output Size	1
Total Regression Outputs	3

(b) Training Configuration

Parameter	Value
Optimizer	Adam
Initial Learning Rate	0.001
Number of Epochs	300
Mini-Batch Size	24

where y_i and $\hat{y}_i$ denote the ground-truth and predicted values of the i-th sample, respectively, and N is the total number of samples.

For the classification task of interference detection, the categorical Cross-Entropy (CE) loss is used, defined as [Goodfellow et al. 2016]:

$$\mathcal{L}_{\mathrm{CE}} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} y_{i,c}\log(\hat{p}_{i,c}), \tag{3}$$

where C is the number of classes, $y_{i,c}$ is the ground-truth label, and $\hat{p}_{i,c}$ is the predicted probability of class c.

Given the heterogeneous nature of regression and classification objectives, the overall multi-task loss is computed as the sum of the individual task losses:

$$\mathcal{L}_{\mathrm{MTL}} = \mathcal{L}_{\mathrm{MSE}} + \mathcal{L}_{\mathrm{CE}}, \tag{4}$$

which corresponds to equal weighting of regression and classification tasks in this work.

Model efficiency is further evaluated in terms of trainable parameters, model size, FLOPs, training and testing time, and average inference latency (see Table 4), assessing suitability for real-time or near–real-time 5G network applications.

Table 4. Model Efficiency Metrics for Evaluated Architectures

Metric	Description
Trainable Parameters	Total number of learnable parameters in the model
Model Size	Disk size of the trained model (MB)
FLOPs	Floating Point Operations per forward pass, indicating computational cost
Training Time	Total time required to train the model
Testing Time	Time required to evaluate the model on the test set
Average Inference Latency	Average time per sample for model prediction, indicating suitability for real-time applications

4.5. Experimental results

The experimental results highlight the inherent trade-offs involved in jointly modeling interference detection and KPI prediction, particularly in terms of classification accuracy, regression performance, and computational efficiency. A quantitative comparison of all evaluated models is reported in Table 5. The results clearly demonstrate that different MTL strategies exhibit distinct strengths, confirming that improvements in one objective may come at the expense of others. Among the single-task learning approaches, STL achieves strong performance in both classification and regression tasks, reaching a classification accuracy of 85.44%. However, this performance comes at a substantial computational cost. As reported in Table 6, STL incurs significantly higher FLOPs (37.84M) and longer training time compared to most multi-task models, due to duplicated feature extraction across tasks. This indicates that while task isolation can be effective in terms of predictive accuracy, it is inefficient for scalable joint modeling. Hard parameter sharing serves as a lightweight multi-task baseline. By enforcing fully shared representations, Hard Sharing achieves the smallest model size (1.52 MB) and the lowest inference latency (0.0017 ms). Nevertheless, the rigidity of complete parameter sharing severely limits the model's ability to handle conflicting task objectives, resulting in the lowest classification accuracy (64.76%) and the highest regression errors across KPIs among all evaluated approaches. More flexible parameter-sharing mechanisms, such as Cross-Stitch networks and PLE, alleviate these limitations by enabling partial task-specific feature specialization. Cross-Stitch substantially improves classification accuracy to 85.31% while maintaining moderate regression losses, demonstrating the benefit of soft feature sharing. However, this improvement is achieved at the cost of increased computational complexity, with FLOPs and training time comparable to STL. PLE further enhances task decoupling through hierarchical expert routing and achieves the highest classification accuracy among all models (87.62%). Despite this gain, the increased model capacity leads to higher computational cost and does not result in consistently lower regression errors across all KPIs. The attention-based multi-task model offers a balanced compromise between accuracy and efficiency. It achieves competitive classification accuracy (85.99%) and relatively low regression losses, while maintaining a compact model size and low inference latency. Although its total test loss is higher than that of MMoE and PLE, the attention mechanism effectively emphasizes task-relevant features, leading to stable and robust joint performance. MMoE demonstrates the most favorable overall trade-off between predictive performance and computational efficiency. As shown in Table 5, MMoE achieves the lowest total test loss (0.2064) among all evaluated models, alongside high classification accuracy (86.80%) and consistently low regression errors across all KPIs. Importantly, this performance is obtained with significantly fewer FLOPs than high-capacity architectures such as STL and Cross-Stitch (9.85M versus approximately 37M), as summarized in Table 6. This confirms that expert-based routing is effective in mitigating negative

transfer while maintaining computational efficiency.

Overall, the results confirm that MTL for interference-aware KPI prediction is inherently trade-off–driven. Architectures that enable adaptive yet controlled task interaction—particularly expert-based models such as MMoE—provide the most favorable balance between accuracy, robustness to task interference, and computational efficiency, which is critical for real-time and self-optimizing 5G radio access network applications.

Table 5. Test performance comparison of single-task and multi-task learning models.

Model	Total Loss	Reg. Loss	Cls. Loss	RSRP	RSRQ	SINR	Acc (%)
MMoE	0.2064	3.29E-04	0.2061	1.54E-04	8.18E-04	1.46E-05	86.80
HardShare	0.3826	9.80E-03	0.3729	1.85E-02	1.07E-02	1.87E-04	64.76
STL	0.2446	1.04E-03	0.2436	5.88E-05	2.98E-03	8.16E-05	85.44
CrossStitch	0.2636	1.50E-03	0.2621	3.52E-04	4.02E-03	1.12E-04	85.31
PLE	0.2170	9.57E-04	0.2161	1.42E-04	2.70E-03	2.89E-05	87.62
Attention	0.2747	6.71E-04	0.2741	2.68E-04	1.70E-03	4.36E-05	85.99

Table 6. Computational cost and model complexity comparison.

Model	Params	Model Size (MB)	FLOPs	Train (s)	Test (s)	Inf. (ms)
MMoE	696k	2.66	9.85M	332.0	0.0097	0.0044
HardShare	399k	1.52	9.56M	213.4	0.0077	0.0017
STL	1201k	4.58	37.84M	425.2	0.0103	0.0051
CrossStitch	1070k	4.08	37.71M	426.7	0.0127	0.0056
PLE	1931k	7.37	11.08M	629.6	0.0133	0.0088
Attention	401k	1.53	14.02M	246.1	0.0088	0.0022

To provide a comprehensive evaluation of the proposed models, we analyzed the trade-offs between predictive performance and computational efficiency using the results reported in Table 5 and Table 6, as summarized in Fig. 8. Here, we defined model complexity as a combination of factors including the number of parameters, computational cost measured in FLOPs, architectural characteristics (e.g., number of layers, type of connections, or multi-branch structures), and the complexity of the training procedure. The comparison confirmed that an increase in model complexity, as defined above, does not necessarily translate into superior predictive performance. With respect to regression performance versus computational cost (Fig. 8(a)), STL and Cross-Stitch exhibited the highest FLOPs due to duplicated feature extraction and additional architectural overhead. However, their regression losses remained comparable to, or higher than, those achieved by more efficient multi-task architectures. In contrast, MMoE achieved consistently low regression losses across all KPIs while requiring substantially fewer FLOPs, indicating more effective utilization of model capacity. PLE also attained low regression loss values, but at the expense of significantly increased model size and training cost. The attention-based model demonstrated competitive regression performance with moderate computational complexity, highlighting the benefit of adaptive feature selection without excessive overhead. Fig. 8(b) illustrated the trade-off between inference latency and regression performance. Hard parameter sharing achieved the lowest inference latency due to its fully shared architecture, but this efficiency came at the cost of poor regression and classification performance. The attention-based model maintained low inference latency while achieving stable regression behavior. MMoE exhibited slightly higher latency than

Hard Sharing and Attention, yet remained computationally efficient and delivered consistently strong predictive performance across tasks. In contrast, STL, Cross-Stitch, and PLE incurred higher inference latency without proportional improvements in regression accuracy. Fig. 8(c) compared classification accuracy against model size. The results demonstrated that high classification accuracy does not require excessively large models. PLE achieved the highest classification accuracy (87.62%) but with the largest model size and highest computational cost. MMoE attained comparably high accuracy (86.80%) with a substantially smaller model footprint, highlighting a more favorable accuracy–complexity trade-off. The attention-based model also delivered competitive accuracy (85.99%) with a compact architecture, while Cross-Stitch achieved similar accuracy at the cost of significantly increased model size. Hard Sharing yielded the lowest accuracy despite having the smallest model.

Overall, these results indicated that MTL for interference-aware KPI prediction was governed by performance–efficiency trade-offs. Among the evaluated architectures, expert-based models—particularly MMoE—provided the most balanced compromise between regression accuracy, classification performance, and computational efficiency. These characteristics were observed to be favorable for scenarios requiring low-latency and resource-efficient operation.

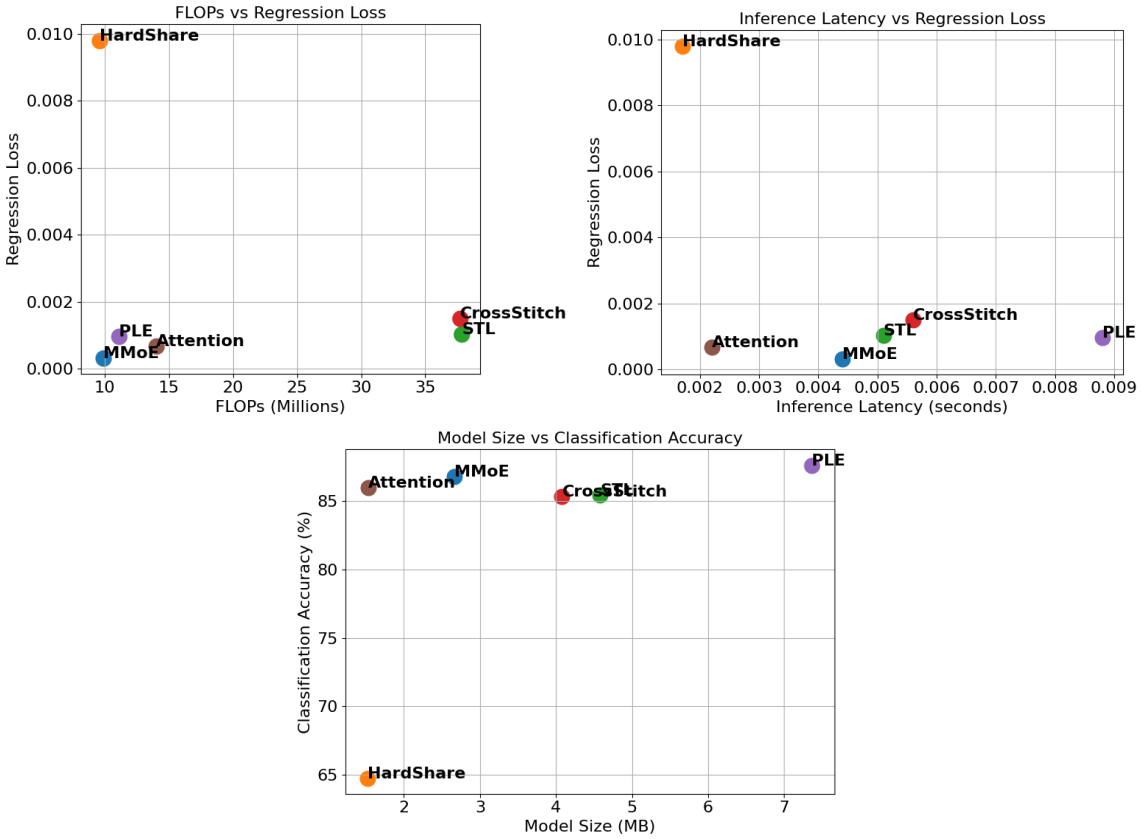

Figure 8. Trade-off analysis between predictive performance and computational efficiency: (a) FLOPs versus regression loss, (b) inference latency versus regression loss, and (c) classification accuracy versus model size.

5. Conclusion

In this paper, we presented a systematic evaluation of MTL architectures for the joint execution of interference detection and radio KPI prediction in 5G networks. By benchmarking strategies ranging from rigid parameter sharing to flexible expert-based routing against a real-world 5G dataset, we demonstrated that the choice of architecture is governed by a critical trade-off between predictive fidelity and computational efficiency.

Our experimental results reveal that heterogeneity matters, as no single architecture achieves optimal performance across all metrics. STL achieves strong predictive performance but incurs high computational cost due to duplicated feature extraction, limiting its practicality for scalable deployment. Hard Parameter Sharing provides a lightweight baseline with minimal inference latency, but its rigid feature sharing leads to poor classification and regression performance. More flexible approaches, such as Cross-Stitch and PLE, improve task interaction and classification accuracy, but introduce substantial computational overhead. In particular, PLE achieves the highest classification accuracy, albeit with the largest model size and training cost. MMoE offers the most favorable balance between accuracy and efficiency. It achieves the lowest total test loss with consistently low regression errors and high classification accuracy, while requiring significantly fewer FLOPs than high-capacity architectures. The attention-based model also delivers competitive performance with low inference latency, but does not outperform MMoE in terms of overall loss–efficiency trade-off. Overall, these results highlight the importance of adaptive yet controlled task interaction in MTL. Expert-based architectures such as MMoE provide an effective and practical solution for interference-aware KPI prediction in real-world 5G radio access networks.

Future work may focus on improving MTL for 5G networks by incorporating adaptive task weighting to reduce negative transfer, extending models with temporal or graph-based representations to better capture spatio-temporal dependencies, and designing lightweight, energy-efficient architectures for real-time deployment. Additionally, evaluating adaptive parameter-sharing mechanisms across heterogeneous network scenarios could provide insights into their generalizability.

Acknowledgments

This work was supported by Ericsson Telecomunicações Ltda., and by the Sao Paulo Research Foundation (FAPESP), grant 2021/00199-8, CPE SMARTNESS.

References

Caruana, R. (1997). Multitask learning. *Machine learning*, 28(1):41–75.

FCCLab (2025). Fcciq dataset 01 – detailed methodology. `https://fcclab.github.io/dataset01.html`. Accessed: 2025-12-14.

Goodfellow, I., Bengio, Y., Courville, A., and Bengio, Y. (2016). *Deep learning*, volume 1. MIT press Cambridge.

Jagannath, A. and Jagannath, J. (2022). Multi-task learning approach for modulation and wireless signal classification for 5g and beyond: Edge deployment via model compression. *Physical Communication*, 54:101793.

Ma, J., Zhao, Z., Yi, X., Chen, J., Hong, L., and Chi, E. H. (2018). Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1930–1939.

Misra, I., Shrivastava, A., Gupta, A., and Hebert, M. (2016). Cross-stitch networks for multi-task learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3994–4003.

Parera, C., Redondi, A. E., Cesana, M., Liao, Q., and Malanchini, I. (2019). Transfer learning for channel quality prediction. In *2019 IEEE International Symposium on Measurements & Networking (M&N)*, pages 1–6. IEEE.

Tang, H., Liu, J., Zhao, M., and Gong, X. (2020). Progressive layered extraction (ple): A novel multi-task learning (mtl) model for personalized recommendations. In *Proceedings of the 14th ACM conference on recommender systems*, pages 269–278.

Tommy, I., Li, X., and Qian, L. (2025). Spatio-temporal beam-level traffic forecasting in 5g wireless systems using multi-task learning. In *2025 IEEE International Conference on Machine Learning for Communication and Networking (ICMLCN)*, pages 1–7. IEEE.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.

Wang, P., Zhang, X., Ma, Y., Zhu, H., Jiao, J., and Zhang, Q. (2024). Mtl-srn: Multi-task learning-based signal recognition network. In *GLOBECOM 2024-2024 IEEE Global Communications Conference*, pages 2918–2923. IEEE.

Zhang, L., Yang, Q., Liu, X., and Guan, H. (2022). Rethinking hard-parameter sharing in multi-domain learning. In *2022 IEEE International Conference on Multimedia and Expo (ICME)*, pages 01–06. IEEE.