CeDA-BatOp 2.0: Enhanced Framework for Base Station Parameter Optimization and Automation with Joint Optimization, Controlled Drift Analysis and Pseudo-Labeling

Sudharshan Paindi Jayakumar, Alberto Conte

CeDA-BatOp 2.0: Enhanced Framework for Base Station Parameter Optimization and Automation with Joint Optimization, Controlled Drift Analysis and Pseudo-Labeling

Sudharshan Paindi Jayakumar*, Alberto Conte
Nokia Bell Labs
Massy, France
*sudharshan.paindi_jayakumar@nokia.com

***Abstract*—In the ever-evolving landscape of cellular networks, the pursuit of optimal network performance remains a constant endeavor. This paper introduces CeDA-BatOp 2.0, an enhanced framework designed to address the complexities of network optimization. Building upon the foundations of its predecessor CeDA-BatOp version 1.0, this upgraded version presents a Multi-task learning (MTL) approach that leverages joint training of a base station parameter predictor and clustering. By intertwining these tasks, CeDA-BatOp 2.0 demonstrates its ability to provide a holistic and efficient solution for network optimization. Furthermore, this framework introduces an innovative analysis of controlled data drift using Gaussian Mixture Models. The controlled drift analysis allows the framework to adapt to real-world data variations effectively, enhancing network performance and reliability. It also incorporates a pseudo-labeling strategy using Multi-View Co-Training, streamlining the retraining process. This feature is crucial in scenarios where manual labeling is impractical, making the system adaptive and efficient in responding to changing network conditions. Through the simulated data, the paper demonstrates the framework's efficacy, achieving significant improvements in network optimization. CeDA-BatOp 2.0 is a step forward in addressing the complexities of cellular network optimization. By combining MTL, drift analysis, and pseudo-labeling, it provides a comprehensive solution that adapts to the dynamic nature of cellular networks, ultimately enhancing network performance and the user experience.**

***Index Terms*—Base station parameter, Network automation, Joint Optimization, Drift Analysis, Pseudo Labeling**

I. Introduction

Artificial intelligence (AI) has assumed an increasingly prominent role within the domain of cellular networks, aimed at enhancing efficiency, optimizing network performance, and elevating the overall user experience. Notably, AI finds one of its pivotal applications in the realm of cellular network resource optimization, a concept substantiated by several comprehensive studies [1], [2]. Empowered by AI and machine learning (ML) algorithms, the scrutiny of extensive datasets originating from diverse network elements, including base stations and user devices, enables the discernment of intricate patterns and the accurate forecasting of network congestion. This in turn equips network operators with the proactive capacity to fine-tune network resources, whether by augmenting bandwidth allocation in high-traffic regions or strategically expanding the number of base stations, as evidenced by research [3], [4].

A remarkable manifestation of AI's influence in cellular networks is observed in the domain of self-organizing networks (SON). These networks, characterized by their remarkable autonomy and minimal human intervention, owe their efficiency to AI algorithms that diligently analyze network data. This analysis enables dynamic adaptations to enhance coverage, capacity, and overall network efficiency, promising significant performance improvements and cost reductions for network operators [5].

Furthermore, the integration of AI technology within cellular networks is geared towards enhancing the quality of the user experience. AI and ML algorithms, for instance, enable the examination of user device data, thus facilitating the personalization of user experiences. These personalized features span from tailored content recommendations to the optimization of network settings for enhanced battery life, in accordance with recent research [6]. Additionally, AI is harnessed to predict and preempt network service disruptions. AI-driven algorithms adeptly scrutinize data from network nodes, promptly identifying potential issues, whether rooted in hardware or software anomalies, allowing network operators to take preventive measures and maintain network reliability, as highlighted by pertinent research [7], [8].

The efficient data transfer and seamless connectivity vital to modern society hinge on the efficacy of these networks. Meeting this challenge necessitates the continuous exploration of innovative solutions that not only adapt to the dynamic nature of cellular networks but also address the intricacies of network optimization. In the contemporary global landscape, marked by highly dynamic base station traffic patterns, the optimization of base station parameters is a pressing necessity. Manual collective optimization within limited regions is not only time-consuming but also reliant on domain experts for parameter adjustments. Overcoming the dearth of measurement data required for accurate conclusions in ML-based optimization problems presents another formidable challenge.

Given the aforementioned challenges, this paper proposes an enhanced framework for base station parameter optimization and automation with Joint Optimization, Controlled Drift Analysis and Pseudo-Labeling that has the potential to significantly impact base station parameter optimization in modern wireless networks. This automated framework can be extended to address other challenges, such as outages, disruptions, and personalized user experiences. Through the seamless integration of this framework as xApps and in the Service Management and Orchestration (SMO) within ORAN's RIC [9], our objective is to enhance the operational capabilities of cellular networks.

Our main contributions include the development of CeDA-BatOp version 2.0, which builds upon the established functionalities of CeDA-BatOp version 1.0 [10]. This upgraded version introduces the following:

1) Utilization of Multi-task learning (MTL) [11] for joint training of base station parameter predictor and clustering to attain the global optimum. By leveraging the interplay between these tasks, our framework enhances its ability to provide a holistic and efficient solution for network optimization. For the predictor, we use Fully Connected Neural Network (FCNN) - 5 layers and Agglomerative clustering (AHC) as the clustering methodology.
2) Incorporating analysis of controlled drift generated using Gaussian Mixture Models (GMM) [12]. It paves the path to an in-depth exploration of the effects of drift intrinsic to the domain of the respective features. By systematically managing and analyzing drift, our framework gains the capability to address real-world data variations and their implications on network optimization.
3) Pseudo-labeling strategy using Multi-View Co-Training (MVCT) [13] to streamline the retraining process. This is a crucial aspect of maintaining the accuracy and relevance of ML models in a dynamic network environment. By combining these strategies, our framework can effectively generate labels for the retraining cycle. This is particularly valuable in scenarios where manual labeling is time-consuming or impractical. It facilitates the retraining process by reducing the reliance on large lookup tables or extensive human intervention. As a result, the system becomes more adaptive and efficient, as it can continuously update its models to respond to changing network conditions and data distributions. For MVCT, we leverage Hyperoptimized Gradient Boosting (HGBoost) proposed in [14] and FCNN – 5 layers.

The subsequent sections are structured as follows: Section II presents an extensive overview of relevant literature. In Section III, the complex components and their stages within the framework are detailed. Section IV elucidates the simulation setup and ML algorithm parameters. A comprehensive analysis of the empirical outcomes demonstrates the framework's automation efficacy. Finally, Section V summarizes the primary findings and implications of the study. It also delves into potential prospects for future research, aiming to deepen and extend the current findings.

II. Related Works

In this section, we present the background for the concepts pertinent to the framework:

Beginning with base station clustering, advancements in this field, particularly in beamforming applications, have demonstrated considerable efficacy, as highlighted in the work by Hong et al. [15] concerning base station clustering and beamforming for partial coordinated transmission in dense urban environments. Similarly, notable contributions, such as the utilization of zero-forcing techniques for intra-cluster transmission, exemplified in [16], have significantly bolstered the efficiency and capacity of base station clustering. Aligning base station clusters with complementary usage profiles, as demonstrated in the work by Chen et al. [17], has optimized processing resources and improved network performance, although its adaptability across diverse network settings remains a challenge, thus highlighting the need for a versatile framework that accommodates various parameter types and diverse network scenarios.

According to Wang et al. [18], who introduced innovative ML techniques for base station design optimization, ML has a significant impact on optimizing base station parameters within self-organizing networks. The integration of Bayesian optimization and reinforcement learning for coverage and capacity optimization by Dreifuerst et al. [19] showcases the importance of multiple input parameters such as throughput, SINR, RSRP, and RSRQ. Our proposed method CeDA-BatOp 1.0 extended this by incorporating a larger set of input parameters, such as RSRQ, RSRP, RSSI, SINR, CQI, DL/UL bitrate, download process status, and neighboring base station parameters to optimize base station parameters effectively. This comprehensive approach is pivotal for effective base station parameter optimization, ultimately ensuring both optimal network performance and an enhanced user experience.

In the domain of data drift within cellular networks, the implications of both data and concept drift are of considerable significance, especially in mission-critical applications. Notably, while the financial sector has extensively harnessed these concepts, their potential in the context of cellular networks has remained underexplored. Predominantly, prior research in this area has concentrated on the detection of anomalies [20], with more recent applications encompassing the identification of shifts in user behavior patterns [21] and changes in network traffic dynamics within the framework of federated learning [22]. CeDA-BatOp v1.0 adopts a proactive approach, continuously monitoring data drift and initiating retraining procedures when necessary to preserve the accuracy of its models. This proactive stance empowers network operators to adapt to evolving network conditions, leading to the ultimate enhancement of network performance and reliability. The integration of data drift detection into cellular networks serves as a catalyst for the timely identification and resolution of potential issues, thereby ensuring the efficiency and dependability of network operation.

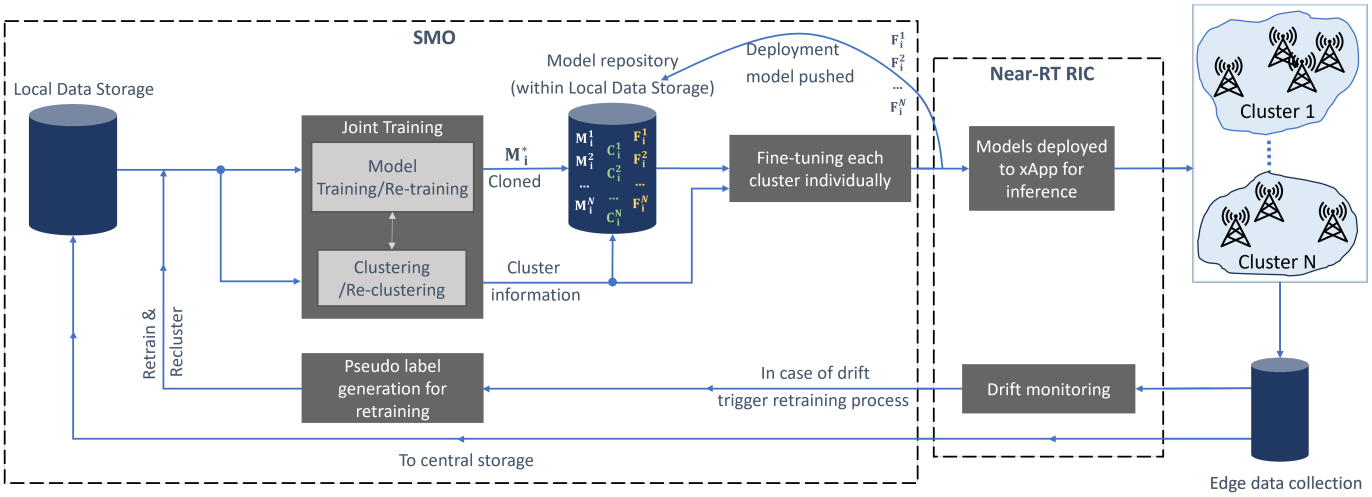

FIG. 1: Base station Parameter Optimization and Automation Framework

In CeDA-BatOp 1.0, the approach involved training the clustering and prediction systems independently, a method that might not consistently lead to achieving the global optimum. Moreover, the drift analysis conducted utilized a method involving random value shifts, resulting in the generation of data values that may not align with the native domain of the respective features. To delve deeper into the effects of drift inherent to the domain of the respective features, there is a need to conduct a controlled drift study. In addition, version 1.0 lacks the incorporation of pseudo-labeling to generate labels for the retraining process. Incorporating pseudo-labeling has the potential to eliminate the need for a large lookup table, thereby improving the system's overall performance. These invaluable insights play a crucial role in influencing the ongoing development of CeDA-BatOp, directing our efforts to improve its resilience and efficacy. They serve as the foundation upon which we envision the emergence of CeDA-BatOp 2.0, symbolizing the evolution of our framework. This paper addresses these limitations, resulting in more impactful and significant outcomes.

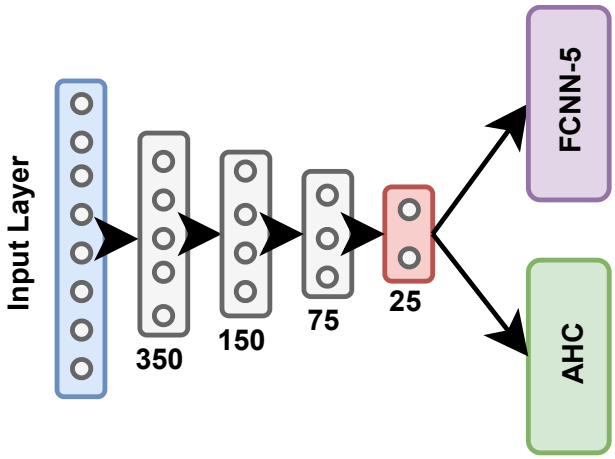

FIG. 2: Joint Training of FCNN-5 and AHC along with the shared encoder

III. FRAMEWORK

Figure 1 shows the proposed base station clustering and parameter optimization framework. The procedure begins with the creation of an offline dataset (D_1) containing UEs and base station parameters. This dataset is generated using an in-house simulator and then stored in a dedicated data repository. The UE and base station parameters serve as the ML model's input and output, respectively. The remainder of the framework consists of four primary stages:

A. Stage 1: Joint training of parameter predictor and clustering

We have an encoder that is shared across the base station parameter predictor (FCNN-5 layers) and clustering task (AHC). Fig. 2 shows our shared representation of the encoder in our pipeline. The encoder section comprises of four layers with the following hidden unit configurations: [350, 150, 75, 25]. Our methodology of training and combining the losses is inspired from Kendall et al. [23]. It is important to note that the scale for losses of clustering and parameter prediction can be different. To overcome this issue, we use the homoscedastic uncertainty method of weighing the losses as proposed in [23]. This is done by learning a noise parameter which is fused into the loss function for each of the individual tasks. As this brings the losses to the same scale, the combined/shared loss which is supposed to be minimized, would now be equivalent to the sum of the individual losses.

For the predictor, we define the likelihood as:

$$p(y|f^W(x),\sigma^2) = \mathcal{N}(y; f^W(x),\sigma^2)$$

where σ^2 is the temperature of the model and σ is also known as the scalar observation noise/uncertainty; x is the input; $f^W(x)$ is the output of the neural network; W represents the weights in the network; y represents the actual output.

For the agglomeration clustering, we define the likelihood as:

$$p(\theta|x,c) = \prod_{i=1}^{n}\prod_{k=1}^{K} \pi_k \mathcal{N}(x_i;\mu_k,\Sigma_k)$$

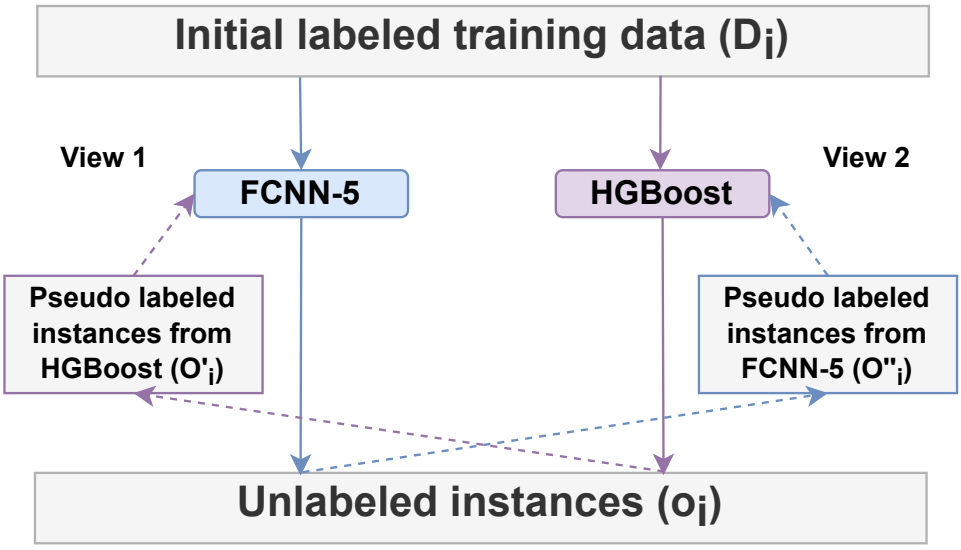

FIG. 3: Multi-View Co-training: FCNN-5 as view 1 and HGBoost as view 2

where θ is the set of parameters of the Gaussian mixture model, including the cluster proportions π_k, the cluster means μ_k, and the cluster covariance matrices Σ_k; n is the number of data points; x_i is the i^{th} data point; c_i is the cluster label of the i^{th} data point; K is the number of clusters; $\mathcal{N}(x; \mu, \Sigma)$ is the probability density function of a Gaussian distribution with mean μ and covariance matrix Σ.

The joint loss of the neural network and the agglomeration clustering system is given by:

$$\begin{aligned}\mathcal{L} &= -log(p(y|f^W(x), \sigma^2)) - log(p(\theta|x, c)) \\ &= \frac{1}{2\sigma^2}||y_i - f^W(x_i)||^2 - log(p(\theta|x, c)) + log\ \sigma\end{aligned}$$

The above loss is jointly minimized by updating the weight W, θ and learning the noise parameter σ.

Initial cycle: The joint loss function is used to train the multi-task model. During training, the shared layers of the neural network will learn features that are beneficial for all tasks, whereas the task-specific layers will adapt to the specific requirements of each task. In our case, we have the neural network model M_1^* which is trained on the initial offline dataset D_1. The training period, which spans E epochs, is set by employing the early stopping mechanism to ensure that the model achieves convergence to an optimal state without suffering from overfitting. In conjunction with this training, base stations are clustered using the pairwise constrained clustering technique as mentioned in CeDA-BatOp v1.0 and data clusters C_1^1,... C_1^N are formed; here subscript 1 refers to the iteration number, initial iteration in this case. This coordinated effort allows the models to collectively achieve the global optimum solution. The pre-trained model M_1^* is then cloned to make M_1^1,... M_1^N for N clusters.

Retraining cycle: The retraining cycle follows a similar strategy as CeDA-BatOp v1.0. The new models, denoted as M_i^1 to M_i^N, where i $>$ 1 and i $\in \mathbb{N}$, are generated by training the pre-trained model M_{i-1}^* on the updated comprehensive dataset $D_i = D_{i-1} \cup O_{i-1}$, for a total of E epochs. These new models are cloned following a similar procedure as previously described. In this context, D_i signifies the overall dataset available for the i^{th} retraining cycle, while O_i represents the newly acquired data (includes the labels) at the i^{th} iteration. Additionally, in the course of joint training, we extract valuable clustered base station information from the dataset D_i.

B. Stage 2: Fine-tuning and Prediction

Utilizing the cluster information obtained through joint training, C_i^1 to C_i^N, where $C_1^1 \subset D_1$,... $C_1^N \subset D_1$ for the initial cycle, and $C_i^1 \subset D_i$,... $C_i^N \subset D_i$ in case of the retraining cycle, we proceed to fine-tune the respective pre-trained models, denoted as M_i^1 to M_i^N. This fine-tuning process involves training these models individually for a specified number of epochs denoted as Y (where Y $<$ E). The resulting fine-tuned models F_N^i,... F_N^i are subsequently stored in the model repository and deployed to the xApp for the inference of base station parameters.

C. Stage 3: Controlled Drift Analysis and Monitoring

The new data, denoted as o_i (input parameters) $\subset O_i$ (input & output parameters) is obtained from the UEs via the base stations and contains the inputs for the Fully Connected Neural Network (FCNN) model. The acquisition occurs at predetermined intervals and is subjected to continuous monitoring for the presence of data drift. In the event of detecting such drift, an automated retraining process is initiated. For this section, we focus on exploring and analyzing the implications and effects of controlled drift generated using GMMs.

D. Stage 4: Pseudo-label generation for retraining process

Once the drift monitoring system triggers the retraining process, pseudo base station parameters (labels) for the corresponding new data are required. We leverage MVCT which considers perspectives of data to improve model performance. It enables the model to learn from diverse views of the

data, enhancing its robustness and accuracy. This approach is particularly beneficial when working with heterogeneous or complex datasets, as it allows the model to uncover patterns and insights that may not be apparent when using a single view. Here we apply MVCT to enhance the predictive power of the retraining module, resulting in more effective and reliable outcomes.

We have two views, FCNN-5 and HGBoost for the MVCT task as shown in Fig. 3. With the Initial dataset D_i available at i^{th} iteration, we train the FCNN-5 and HGBoost. For the initial cycle of these views, we leverage the structure of the FCNN-5 model M_1^* from Stage 1, but with the altered input layer to take into account the reduced input features as shown in Table IV. With the trained model of FCNN-5 and HGBoost, we now infer the labels for the new unlabelled data o_i which provides the intermediate pseudo-labels. With the intermediate pseudo-labeled instances of HGBoost (O_i'), we now train the FCNN-5, and similarly, with the intermediate pseudo-labeled instances of FCNN-5 (O_i''), we train the HGBoost. The number of training cycles for MVCT would depend on the availability of computational time and the risk of overfitting if trained for many cycles.

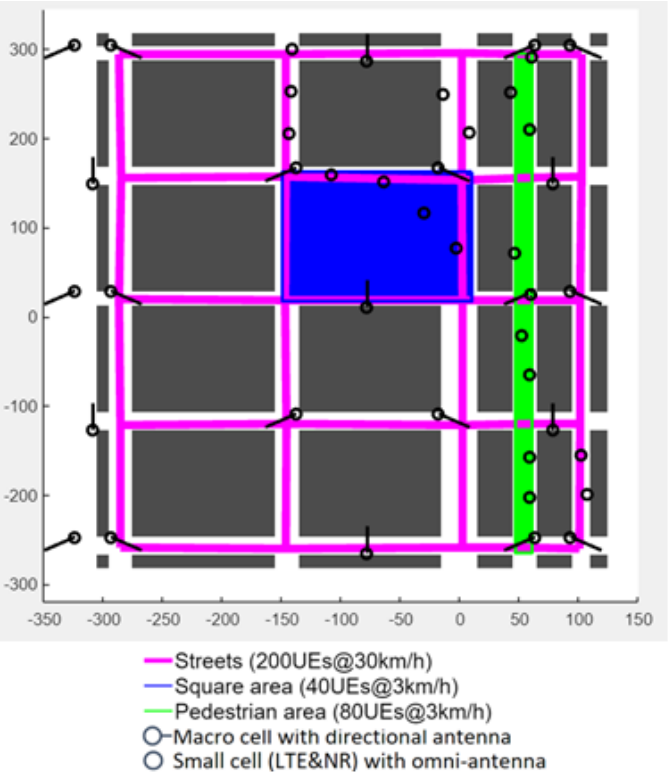

FIG. 4: Map of a simulated locality in Madrid

IV. Simulation Setup, Discussion and Result

In this study, we use the same meticulously curated simulated dataset that was used in CeDA-BatOp v1.0. This local dataset, collected using a proprietary dynamic system-level simulator, comprises 42 strategically placed base stations, accurately reflecting the urban setting of a specific region in Madrid. Within this dataset, a diverse range of base station configurations is represented, encompassing 23 macro cells equipped with directional antennas and 19 small cells employing Omni-antennas. The mobility patterns of UEs are modeled across varying speeds, including 200 UEs moving at 30 km/hr, 40 UEs at 3 km/hr, and 80 UEs at 3 km/hr. This diversified representation of UE movement speeds introduces realistic mobility patterns, thereby enabling the ML model to adapt to a spectrum of user scenarios. The amalgamation of diverse scenarios, movement velocities, and base station configurations positions it as a highly valuable resource for the development and validation of base station-specific ML models. These models are aimed at the optimization of network performance and the enhancement of the overall user experience in urban environments like Madrid. Please see Fig. 4 for a visual representation of the simulated scenario.

The dataset encompasses a rich repository of UE parameters: Reference Signal Received Quality (RSRQ), Reference Signal Received Power (RSRP), Reference Signal Strength Indicator (RSSI), Signal-to-interference-plus-noise ratio (SINR), Channel quality indication (CQI), Download/upload bitrate (DL/UL bitrate), State (indicating whether the UE is in an idle or downloading state), Reference Signal Received Quality of the neighboring cell (NRxRSRQ) and Reference Signal Received Power of neighboring cell (NRxRSRP). This comprehensive dataset also includes optimized base station parameters: Antenna tilt (comprising both Mechanical tilt and Electrical tilt), Antenna azimuth and Maximum Transmission power. Each parameter within the dataset encompasses a rich set of features. For instance, in the case of RSRQ, the dataset encapsulates RSRQ values across various time steps for all 42 base stations. In total, the dataset consists of 736 input features at the UE end and 4 output features at the base station end. A preprocessed sample of this dataset is available here.

For stage 1 of the framework, we leverage FCNN-5 (12,25,12,10) and AHC. These were chosen based on the results from CeDA-BatOp v1.0: On our dataset, FCNN-5 provided the best trade-off between training time and Root Mean Square Error (RMSE), AHC obtained the highest Silhouette score [24]. The hyperparameters search can be inferred from [10].

The flow of training and prediction in the case of Multi-task training is as follows: We first train the FCNN-5 independently on the overall dataset and the resulting model is stored in the model repository. We call this initial training. RMSE obtained is 28.34. With AHC run independently, we clustered the base stations and using the method of dendrograms [25], we found the best cluster size to be 9 and obtained a Silhouette Score of 0.779. These have been detailed in CeDA-BatOp v1.0. We now train both these models using the MTL joint loss function. The obtained results of AHC and FCNN-5 in comparison to that of CeDA-BatOp v1.0 for cases with and without initial training are tabulated in Table I and Table II respectively. It is interesting to notice that for the same cluster size, AHC has an avg. silhouette score of 0.853 and can now form better clusters when trained jointly compared to training independently. In

addition, we obtain the best RMSE of 17.62 for the case when initial training of FCNN is performed. Unless explicitly mentioned, it is presumed that parameters not specified have been set to their default configurations, as offered by the scikit-learn library [26]. The experimental results were generated using a Samsung 970 EVO for data storage and retrieval, while the training was conducted on an Intel Core i7-10750H processor.

TABLE I: AHC Comparison for CeDa-BatOp versions 1.0 and 2.0

Clustering Algorithm	Version	Avg. Silhouette Score
AHC	CeDA-BatOp 1.0	0.779
	CeDA-BatOp 2.0	**0.853**

As outlined in CeDA-BatOp v1.0, during the fine-tuning phase, we fine-tuned the 5-layer FCNN models, M_1^i,... M_N^i, using their corresponding cluster data, C_1^i,... C_N^i. The fine-tuning process resulted in an RMSE of 16.98. For CeDA-BatOp v2.0, we fine-tuned the final FCNN-5 model obtained through MTL and the results are tabulated in Table III. FCNN-5 with no initial training fetched an RMSE value of 12.75 and 12.34 for FCNN-5 with initial training. We notice a considerable reduction in the RMSE value for FCNN-5 with initial training as compared to CeDA-BatOp v1.0. This leads us to conclude that fine-tuning the pre-trained model customizes the prediction process for the respective clusters, thereby resulting in improved predictions. The fine-tuned models obtained above can subsequently be deployed as xApps for making inferences on optimized base station parameters.

TABLE II: Performance of FCNN-5 (12,25,12,10) when trained on the overall dataset with initial training (without cluster fine-tuning)

Model	Version	Initial Training	RMSE
FCNN-5 (12,25,12,10)	CeDA-BatOp 1.0	Not Applicable	28.34
	CeDA-BatOp 2.0	No	19.89
		Yes	**17.62**

TABLE III: Performance of FCNN-5 (12,25,12,10) when trained on the overall dataset with initial training and cluster fine-tuning

Model	Version	Initial Training	RMSE
FCNN-5 (12,25,12,10)	CeDA-BatOp 1.0	Not Applicable	16.98
	CeDA-BatOp 2.0	No	12.75
		Yes	**12.34**

To investigate the impact of drift within the realm of UE parameters' domain, we conducted a controlled drift experiment utilizing Gaussian Mixture Models (GMMs). Note that we drift only 50% of the overall dataset to study the drift system. Initially, we scaled down the raw data by a factor of two, ensuring that the standard deviation of the newly derived data equated to 50% of the original data's standard deviation. We then employed classical GMMs to estimate an unimodal distribution for this scaled data by setting the number of Gaussian mixtures to 1. The noisy data is then sampled from the respective unimodal distribution and then added to the corresponding data in such a way that the signal-to-noise ratio (SNR) ≈166%. In other terms, this means that for every 100 units of signal, there were 60 units of noise introduced. This process was repeated for a range of Gaussian mixtures, from 1 (representing an unimodal distribution) to 10 (representing a multimodal distribution). We then use similarity tests - Kolmogorov-Smirnov (KS) test [27], Population Stability Index (PSI) [28] and Jensen-Shannon divergence (JS) [29] to identify and quantify the presence of drift within this new dataset as detailed in CeDA-BatOp v1.0 and also use the same thresholds. From v1.0, we noticed that Consensus Average Accuracy (CAA), a majority voting approach provided the best accuracy on our dataset. For example, if the KS test confirms the presence of drift, PSI suggests no drift, and JS indicates drift, we conclude that drift has occurred. This decision is based on a voting system, where two algorithms support drift, and one opposes it.

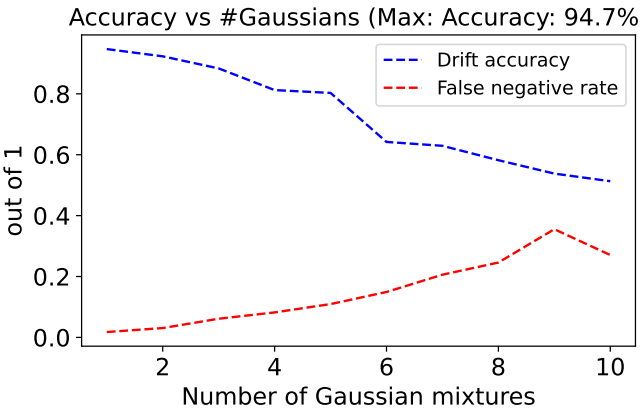

FIG. 5: Drift detection accuracy across GMMs for SNR = 166%

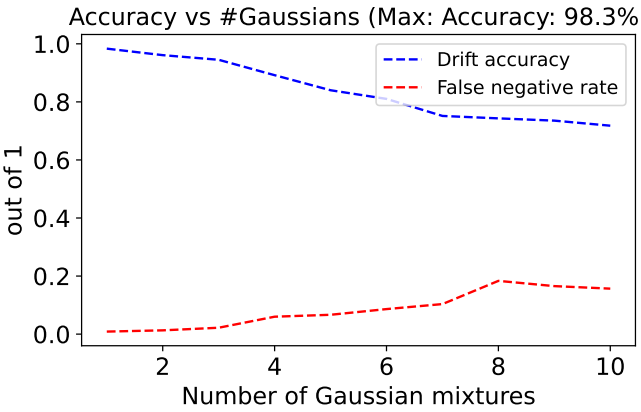

FIG. 6: Drift detection accuracy across GMMs for SNR = 125%

Fig. 5 shows the results obtained for the drift detection system for SNR ≈ 166% with different Gaussian mixtures. We obtain a maximum drift accuracy of 94.7% for the case when noise is sampled from unimodal distribution. It is also evident that an increase in the number of Gaussian mixtures leads to a decline in the accuracy of the drift detection system. With an elevated number of Gaussian mixtures, the distribution closely approximates the actual data, and the noise sampled

from this distribution can resemble the true data, making it less distinguishable as a drift from the original distribution. This trend is also noticeable in the rising false negative rate, which escalates as the number of mixtures increases.

Furthermore, we extended our investigation to SNR $\approx 125\%$ (indicating 80 parts of noise for every 100 parts of signal). Similar to the previous case, we explored various numbers of Gaussian mixtures, ranging from 1 to 10. The outcomes are visualized in Figure 6. The highest accuracy achieved is 98.3%, which corresponds to the case when noise is sampled from an unimodal distribution, similar to the previous scenario. We can also observe a steady drop in the accuracy as we increase the number of mixtures in GMM. It is worth highlighting that in the case of SNR $\approx 166\%$, the accuracy exhibits a linear decrease, whereas in the case of SNR $\approx$ 125%, it approaches a saturation point as the number of mixtures increases. This exploratory analysis illuminates the performance of the drift detection system in scenarios where the detected drift closely mirrors the inherent variability within the overall data distribution. In cases where the introduced noise from data drift aligns subtly with the natural data variance, it can diminish the sensitivity and accuracy of the drift detection system. This observation underscores the challenges faced by the system in identifying minor drift instances or when such variations seamlessly blend with the underlying natural data distribution. Whether this performance aligns with specific KPI requirements is contingent upon the particular needs and objectives.

TABLE IV: Ranked based on Permutation Importance score (for initial cycle of the framework)

Rank	FNN-5	HGBoost
1	RSRP	RSRQ
2	RSSI	NRxRSRQ
3	CQI	NRxRSRP
4	NRxRSRQ	RSRP
5	SNR	SNR
6	NRxRSRP	DL bitrate
7	RSRQ	State (I/D)
8	UL bitrate	CQI
9	DL bitrate	UL bitrate
10	State (I/D)	RSSI

Once the drift is detected, we trigger the retraining process. We considered two views, implying we use two algorithms to capture two different perspectives which will later complement each other. We have two regressors - FCNN-5 (12,25,12,10) and HGBoost. The MVCT algorithm is based on the belief that the input features of these two regressors are disjoint. In order to split the features, we need to obtain the rank of the features for both regressors. This is obtained using Permutation Importance score [30] and is represented in Table IV. The overall dataset D_i is now split into two feature sets - Feature Set Y ($D_i{}^Y$) containing the top 5 ranks of FCNN-5 and Feature Set Z ($D_i{}^Z$) containing the top 5 ranks of HGBoost. $D_i{}^Y$ is then fed into the FCNN-5 thereby obtaining a trained model. The new unlabelled instances o_i are then inferred on this FCNN-5 to obtain the labels for the unlabelled instances. Likewise, this procedure is applied to HGBoost using $D_i{}^Z$. Following this, the pseudo-labels provided by FCNN-5 are used to train HGBoost and vice-versa. Through this approach, the models acquire distinct knowledge from features that can sometimes be contradictory but collaboratively contribute. This synergy between models streamlines the retraining process for subsequent cycles.

TABLE V: Model performance for pseudo-labeling with MVCT

Model	RMSE
FCNN-5 (12,25,12,10)	21.43
HGBoost	27.64
Prediction mean from FCNN-5 and HGBoost	**19.77**

To assess the effectiveness of the pseudo-label generation system, we calculated the RMSE values by comparing the pseudo base station parameters with the withheld simulated data, which contains the true BS parameters corresponding to the UE features. In our experimental setting, it was observed that after conducting 4 cycles of MVCT training, the FCNN-5 model began to exhibit signs of overfitting. In real-world scenarios where actual labels are unavailable to monitor overfitting, it is advisable to consider a lower number of MVCT cycles as a precautionary measure to ensure the model's generalization and reliability. After the training cycles of MVCT, the resulting models of FCNN-5 and HGBoost are then used to infer the pseudo base station parameters for the next retraining cycle. The results obtained are tabulated in Table V. We can decipher that FCNN-5 has a lower RMSE of 21.43 compared to that of HGBoost. It is intriguing to note that the lowest RMSE of 19.77 is achieved by utilizing the mean of the predictions from the FCNN-5 and HGBoost models. This finding highlights the potential efficacy of a combined approach in improving predictive accuracy.

V. Conclusion

CeDA-BatOp 2.0 is a new framework for base station clustering and parameter optimization that introduces three key improvements over CeDA-BatOp 1.0: multi-task learning for joint training of base station parameter predictor and clustering, analysis of controlled drift generated using Gaussian Mixture Models, and pseudo-labeling strategy using Multi-View Co-Training. Evaluation of joint training in CeDA-BatOp 2.0 on a simulated dataset showed that it outperforms CeDA-BatOp 1.0 in terms of both clustering and prediction performance, with the best results being a Silhouette score of 0.853 and an RMSE of 12.34. The controlled data drift analysis is a crucial component of our in-depth drift analysis, providing valuable insights into the performance of drift detection when drift occurs within the domain of the initial dataset. This exploration enables CeDA-BatOp 2.0 to seamlessly adjust to the ever-changing and unpredictable variations in real-world network data, all while streamlining the retraining process. The integration of a pseudo-labeling strategy using MVCT enhances the retraining process by efficiently generating labels for unlabeled data, thus reducing the need for manual

intervention and extensive lookup tables. This makes the system more adaptive and efficient, continuously updating its models to respond to changing network conditions and data distributions. In summary, CeDA-BatOp 2.0 outlines a clear path toward establishing a more autonomously functioning network infrastructure

Our future work will focus on the investigation of reinforcement learning techniques to enhance dynamic network management within this framework. Additionally, we will delve into the development of an intelligent weighting system to refine the final prediction derived from the dual perspectives in the MVCT pseudo-labeling system.

ACKNOWLEDGMENT

This work is supported by the European Union's Horizon 2020 research and innovation programme through the SEMANTIC project (Grant No. 861165).

REFERENCES

[1] Janne Riihijarvi and Petri Mahonen. Machine learning for performance prediction in mobile cellular networks. *IEEE Computational Intelligence Magazine*, 13(1):51–60, 2018.

[2] Chunxiao Jiang, Haijun Zhang, Yong Ren, Zhu Han, Kwang-Cheng Chen, and Lajos Hanzo. Machine learning paradigms for next-generation wireless networks. *IEEE Wireless Communications*, 24(2):98–105, 2016.

[3] Chaoyun Zhang, Paul Patras, and Hamed Haddadi. Deep learning in mobile and wireless networking: A survey. *IEEE Communications surveys & tutorials*, 21(3):2224–2287, 2019.

[4] Kun Yang, Cong Shen, and Tie Liu. Deep reinforcement learning based wireless network optimization: A comparative study. In *IEEE INFOCOM 2020-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 1248–1253. IEEE, 2020.

[5] Olav Østerbø and Ole Grøndalen. Benefits of self-organizing networks (son) for mobile operators. *Journal of Computer Networks and Communications*, 2012, 2012.

[6] Nanticha Poonpanich and Jiroj Buranasiri. Factors affecting baby boomers' attitudes towards the acceptance of mobile network providers' ai chatbot. *Jurnal Nasional Pendidikan Teknik Informatika: JANAPATI*, 11(3):176–182, 2022.

[7] Deniz Gündüz, Paul de Kerret, Nicholas D Sidiropoulos, David Gesbert, Chandra R Murthy, and Mihaela van der Schaar. Machine learning in the air. *IEEE Journal on Selected Areas in Communications*, 37(10):2184–2199, 2019.

[8] Yeh-Hong Ping and Po-Chiang Lin. Cell outage detection using deep convolutional autoencoder in mobile communication networks. In *2020 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 1557–1560. IEEE, 2020.

[9] ORAN Community. Ric applications (ricapp), Sep 2022.

[10] Sudharshan Paindi Jayakumar and Alberto Conte. Framework: Clustering-Driven Approach for Base Station Parameter Optimization and Automation (CeDA-BatOp). Accepted in IEEE Consumer Communications & Networking Conference (CCNC), January 2024. https://hal.science/hal-04201134.

[11] Rich Caruana. Multitask learning. *Machine learning*, 28:41–75, 1997.

[12] Douglas A Reynolds et al. Gaussian mixture models. *Encyclopedia of biometrics*, 741(659-663), 2009.

[13] Chang Xu, Dacheng Tao, and Chao Xu. A survey on multi-view learning. *arXiv preprint arXiv:1304.5634*, 2013.

[14] Erdogan Taskesen. hgboost is a python package for hyperparameter optimization for xgboost, catboost and lightboost for both classification and regression tasks., October 2020.

[15] Mingyi Hong, Ruoyu Sun, Hadi Baligh, and Zhi-Quan Luo. Joint base station clustering and beamformer design for partial coordinated transmission in heterogeneous networks. *IEEE Journal on Selected Areas in Communications*, 31(2):226–240, 2013.

[16] Quentin H Spencer, A Lee Swindlehurst, and Martin Haardt. Zero-forcing methods for downlink spatial multiplexing in multiuser mimo channels. *IEEE transactions on signal processing*, 52(2):461–471, 2004.

[17] Longbiao Chen, Dingqi Yang, Daqing Zhang, Cheng Wang, Jonathan Li, et al. Deep mobile traffic forecast and complementary base station clustering for c-ran optimization. *Journal of Network and Computer Applications*, 121:59–69, 2018.

[18] Subin Wang, Qi Wu, Chen Yu, Haiming Wang, Wei Hong, and Weishuang Yin. Dual-polarized base station antenna design using machine learning-assisted optimization method. In *2021 IEEE International Symposium on Antennas and Propagation and USNC-URSI Radio Science Meeting (APS/URSI)*, pages 1715–1716. IEEE, 2021.

[19] Ryan M. Dreifuerst, Samuel Daulton, Yuchen Qian, Paul Varkey, Maximilian Balandat, Sanjay Kasturia, Anoop Tomar, Ali Yazdan, Vish Ponnampalam, and Robert W. Heath. Optimizing coverage and capacity in cellular networks using machine learning. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8138–8142, 2021.

[20] Rosana Noronha Gemaque, Albert França Josuá Costa, Rafael Giusti, and Eulanda Miranda Dos Santos. An overview of unsupervised drift detection methods. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 10(6):e1381, 2020.

[21] Dimitrios Michael Manias, Ali Chouman, and Abdallah Shami. A model drift detection and adaptation framework for 5g core networks. *arXiv preprint arXiv:2209.06852*, 2022.

[22] Amir Hossein Estiri and Muthucumaru Maheswaran. Attentive federated learning for concept drift in distributed 5g edge networks. *arXiv preprint arXiv:2111.07457*, 2021.

[23] Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7482–7491, 2018.

[24] Peter J Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65, 1987.

[25] M Forina, C Armanino, and V Raggio. Clustering with dendrograms on interpretation variables. *Analytica Chimica Acta*, 454(1):13–19, 2002.

[26] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[27] Giovanni Fasano and Alberto Franceschini. A multidimensional version of the kolmogorov–smirnov test. *Monthly Notices of the Royal Astronomical Society*, 225(1):155–170, 1987.

[28] Bilal Yurdakul. *Statistical properties of population stability index*. Western Michigan University, 2018.

[29] ML Menéndez, JA Pardo, L Pardo, and MC Pardo. The jensen-shannon divergence. *Journal of the Franklin Institute*, 334(2):307–318, 1997.

[30] Leo Breiman. Random forests. *Machine learning*, 45:5–32, 2001.