Experimentation of online models for scalability of 5G network functions

Abrahão Ferreira, Kauan Tavares, Douglas Vidal, Silvia Lins,
Aldebaro Klautau, Cristiano Bonato, and Glauco Gonçalves

***Abstract*— The increasing evolution of 5G networks and the growing demands of new devices pose significant challenges to the proactive scalability of network functions. Deployed in a non-stationary environment, the performance of conventional predictive models drops as new concepts appear, making it necessary to use techniques as online learning, which are still underexplored. This work evaluates and compares fifteen online learning strategies for the scalability of the Access and Mobility Management Function (AMF). This paper contributes to identifying more effective approaches using real data in scenarios with concept drift. Results indicate that model efficacy is contingent on predictive accuracy and adaptation latency.**

***Keywords*— 5G Core, Online learning, Proactive scaling.**

I. Introduction

Current mobile networks face the challenge of meeting the demands of new applications, such as autonomous vehicles and massive communications. These challenges, despite their differing requirements [1], pose a common threat to the network, potentially disrupting your work by coping with an increasing number of devices. Increasing control traffic can cause congestion in the control plane, leading to service provisioning failures. In this sense, the correct dimensioning of network functions at the *5G core* (5GC) is essential to avoid service interruptions [2].

Figure 1 shows a conceptual view of a scalability system for the 5GC. Focusing on the control plane, the figure indicates that control traffic is sent from the *User Equipment* (UE) to the 5GC, through the base station (gNodeB). At its core, the traffic is received by the *Access and Mobility Management Function* (AMF), which is responsible for establishing UE sessions and serves as the gateway to other 5GC functions. In a scalable 5GC, AMF replicas are executed on a network function set, and the traffic is distributed between replicas by a Load Balancer. Furthermore, the Scaling Module scales instances horizontally according to the traffic demand.

Two main strategies can be used for scaling: reactive and proactive. Reactive strategies use current traffic demand and thresholds to create or remove instances. Despite its simplicity and versatility, the time it takes between decision-making and the effective instance creation/removal can lead to service oscillations, i.e., an instance being recreated after its removal. Conversely, proactive methods use predictive models to anticipate demand. Therefore, the Scaling Module makes decisions based on information provided by a Forecasting System, usually a machine learning model, which predicts future demands from collected data traffic. Thus, the method anticipates instance creation/removal, avoiding oscillations.

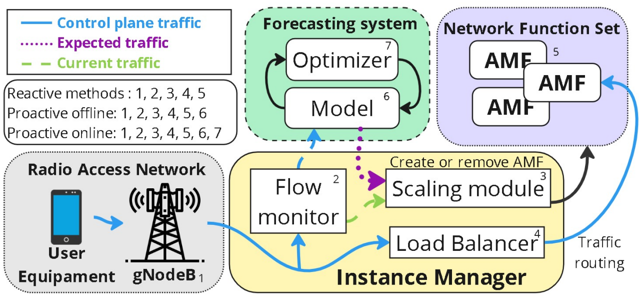

Fig. 1: Conceptual view of a scalable 5GC.

Commonly, proactive methods employ an offline learning approach, where the model is trained using past data and then deployed in the Forecasting System. However, telecommunication networks are non-stationary environments, i.e., the traffic is subject to concept drifts (changes in the statistical distribution of data over time) as new applications and technologies are introduced. This leads to the model's quality degradation, which becomes less effective as the network evolves. In contrast, online machine learning models excel in nonstationary scenarios due to their incremental learning capability. The Optimizer analyzes the model output and continuously improves it for each predicted instance. Due to this characteristic, online models have been widely used to solve problems in non-stationary scenarios. For example, a study [3] obtained promising results investigating online models to scale resources in edge computing environments.

This paper presents an evaluation of online learning models for the proactive scalability of the AMF function in a 5GC. The choice of AMF is justified by its central role in user and session management, which makes it particularly sensitive to traffic variations. Although the study focuses on this function, the method employed could be adapted to other 5GC functions with similar architectural and operational characteristics. The main contribution of this paper is to take an experimental approach to compare the performance of different online learning models in scenarios with non-stationary traffic. The evaluations follow a method defined in this paper, using real

Abrahão Ferreira, Kauan Tavares, Douglas Vidal, Aldebaro Klautau, and Glauco Gonçalves are with Universidade Federal do Pará (UFPA), Belém, Brazil. Silvia Lins is with Ericsson Research, Brazil. Cristiano Bonato is with Universidade do Vale do Rio dos Sinos (UNISINOS), Brazil. Emails: [abrahao.ferreira@itec.ufpa.br, kauan.tavares@itec.ufpa.br, douglas.vidal@itec.ufpa.br, silvia.lins@ericsson.com, aldebaro.klautau@itec.ufpa.br, cbboth@unisinos.br, glaucogoncalves@itec.ufpa.br].

data from a telecommunications operator.

The remainder of this paper is organized into four sections. Section II presents the main work related to the scalability of the 5GC. Section III details the methodology adopted in this work, describing the data used, the predictive models used, and the simulation strategy adopted. Finally, Section IV presents and analyzes the results obtained, followed by the conclusions and future perspectives in Section V.

II. Related work

Some studies have investigated machine learning techniques for proactive 5GC scaling. Authors in [4] use deep neural networks and *Long Short-Term Memory* (LSTM) models to predict demand for scaling 5GC functions. The task is approached as a classification problem, subdividing traffic into load bands, where each band is associated with the number of AMF instances allowed to support it.

The works [5] and [2] investigate the use of convolutional neural networks and *Gated Recurrent Unit* (GRU), for the scalability of AMF functions in Kubernetes environments, by predicting CPU usage. These works simulate the behavior of AMF through a *Web* server with HTTP requests. Moreover, using Kubernetes, authors in [6] investigate the scalability of the AMF using the Wavenet model. The authors implemented an *open source* 5G core, which enabled the analysis of operational metrics.

Concerning online learning approaches, in [7], the authors propose addressing concept drifts by retraining an offline learning model. They use a method to detect drifts and a strategy to accelerate the retraining process. Although it offers a solution to deal with changes in data patterns, retraining approaches are not well-suited to all types of concept drift [8]. In the opposite direction, a previous study [9] by the authors of this paper compared an online learning algorithm with an offline learning algorithm for scaling AMF instances. These initial results suggest that online learning can be crucial for improving predictions in the presence of concept drifts, particularly when the most significant changes occur.

Although the proactive scaling of 5GC functions is not a new problem, the application of online learning to address this issue is still in its early stages. However, a large number of online learning strategies are available in the literature, and the question of which methods are most suitable for this task remains open. This work addresses this gap by evaluating and comparing multiple online learning models for proactive AMF scaling in a non-stationary environment, contributing to the understanding of which strategies are better suited to this task.

III. Evaluation method

Figure 2 illustrates the employed method. The first step, described in Section III-A, involves processing a real mobile network traffic dataset, which ensures that the probability distributions learned by the models are similar to those found in practice while, at the same time, due to the data structure, enabling the introduction of concept drifts. The next step (Section III-B) consists of selecting, training, and evaluating the online learning models for traffic prediction. Finally, Section III-C describes the construction of a simulator to quantify the impact of the models' forecasts on the scalability of the AMF function through high-level operational indicators. All experiments, code, and processed data are publicly available for reproducibility and future research[1].

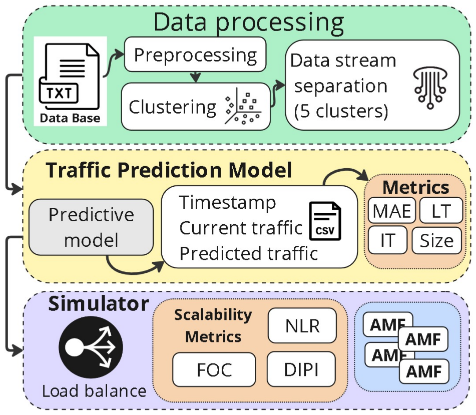

Fig. 2: Evaluation method used in this work.

A. Dataset and Processing

The data used in the evaluation is derived from the collection of telecommunication *Call Detail Records* (CDRs) from Telecom Italia's mobile network users in the Milan city region and its surroundings [10]. The data was collected between November 1, 2013, and January 1, 2014[2], and has been widely used in the mobile network literature [7], [11], [12]. The area is organized as a 100x100 square of cells. In each cell, the CDRs are obtained at 10-minute intervals, containing the timestamp, cell ID, phone calls, SMS, and Internet activities. These CDRs were rescaled by a factor known only to the provider to create a privacy level while maintaining the real traffic distribution.

In this work, following [11], activity data (SMS, phone calls, and Internet) were aggregated into a single measure, denoted Total Activity. Further, following [4], 10% of the Total Activity is considered to correspond to the Control Activity, indicating the number of requests sent to the AMF every 10 minutes. Given the interest in evaluating the models under concept drifts, we assume that different cells in the region may have diverse traffic patterns. Hence, the Control Activity was used to divide the cells into clusters. This way, it is possible to depict abrupt concept drifts while training a model with one cluster of data and using data from a statistically different cluster during testing.

To make the *clusters*, firstly, a position `(x, y)` was assigned to each cell concerning its spatial location. After that, similar to [11], [12], *K-means* algorithm was applied, dividing the cells into groups according to their Control Activity and position. The number of clusters was set to five, following [12]. Figure 3 shows the clusters on the left, and the Cumulative Distribution Function of Control Activity in each cluster (in requests per 10 minutes), on the right.

[1]Available at: `https://github.com/lasseufpa/5g-online-scaling-exp`

[2]Hosted at `https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/EGZHFV`

In the grid, it is notable that Cluster 4 primarily covers the central region of the city, surrounded by Cluster 3. Moreover, Clusters 0, 1, and 2 display comparable areas, with slightly similar distributions, more concentrated in lower values. These findings indicate notable differences in traffic patterns between Clusters 3 and 4 and the others, whereas Clusters 0, 1, and 2 demonstrate a higher degree of similarity among themselves.

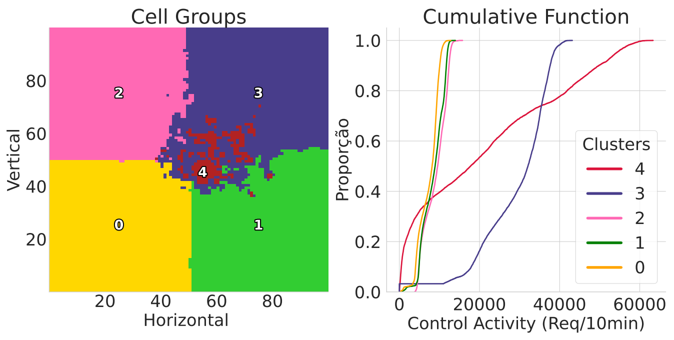

Fig. 3: Left: Grid of cells subdivided into five clusters. Right: Cumulative function of Control Activity in each cluster.

B. Control Activity Prediction Models

Fifteen online learning models were evaluated in this work: *Aggregated Mondrian Forests Regressor* (AMFR), *Adaptive Random Forest Regressor* (ARFR), *Bagging Regressor* (BGR), *Bayesian Linear Regression* (BLR), *Exponentially Weighted Average regressor* (EWAR), *Hoeffding Adaptive Tree Regressor* (HATR), *Hard Sampling Regressor* (HSR), *Hoeffding Tree Regressor* (HTR), *K-Nearest Neighbors Regressor* (KNNR), *Linear Regression* (LR), *Online Extra Trees Regressor* (OXTR), *Passive-Aggressive Regressor* (PAR), *Stochastic Gradient Tree Regressor* (SGTR), *Scikit-learn to River Regressor* (SKL2RR), and *Streaming Random Patches Regressor* (SRPR).

Each model was trained with the Control Activity data from November in cluster 0, adopting the warm start strategy. Therefore, the models begin the evaluation already exposed to a specific traffic pattern, allowing for the investigation of their ability to adapt to changes using data from other clusters. After training, each model was tested using data from December from clusters 0 to 4. December's data from Cluster 0 represents a scenario with only slightly different patterns associated with the end-of-year festivities, and with no concept drifts concerning the training set. In contrast, December's data from cluster 3 is a significant change, due to higher data flow and seasonal variations, making it a more challenging case.

The models use a window of 20 past instances of activity values to forecast the next activity. This value was defined based on the partial autocorrelation analysis applied to the November data, covering the different clusters. The evaluation of the models' performance uses the *Mean Absolute Error* (MAE), a measure of the average magnitude of the errors and is expressed as MAE $= \frac{1}{n}\sum i = 1^n |y_i - \hat{y}i|$. In addition, the computing costs of the models were analyzed according to: *Inference Time* (IT), the mean time required for the model to make a prediction; *Learning Time* (LT), the average time the model takes to update its parameters for each new data sample; and the size of the model in memory, measured in kilobytes (KB) at the end of each experiment.

C. Simulation

The final step of the evaluation involves a trace-based simulation [13] that assesses the various models for provisioning AMF instances and calculates high-level operational indicators. The simulator, developed in Python with the Simpy library[3], receives as input a trace containing the actual Control Activity data and the predicted Control Activity - coming from the model under evaluation - at each instant in time, with a sampling interval of 10 minutes (see Section III-A).

Models of the Scaling Module, Load Balancer, and AMF replicas were implemented. During each time interval, each AMF replica receives a portion of the control traffic from the Load Balancer, processes the amount supported by its capacity, and rejects any exceeding requests. Based on the measurements reported in [4], the capacity of each AMF instance is set to 20 requests per second (i.e., 1,200 requests over 10 minutes).

To minimize the Load Balancer's influence on scaling decisions, the traffic distribution follows the round-robin algorithm, with a slight modification when an instance is being removed. In a 5GC, one should avoid terminating an AMF instance if there are any UEs associated with it. The behavior of real UEs was emulated using observations from [14], which, based on data from an American operator, showed that most UEs remain in service for up to 2 minutes, with only 1% remaining for more than 16 minutes. Therefore, instances are removed gradually: in the first 10-minute interval after the removal decision, the Load Balancer sends traffic to the instance occupying 5% of its capacity; in the next 10 minutes, the share is reduced to 1% before complete removal.

The Scaling Module decides, using Equation 1 at each time interval, the number of instances ($N_{\text{predicted}}$) required to meet the predicted demand. The occupancy of the instances is limited to 80% so that they remain in a region of efficient operation, avoiding reaching prohibitive response times [13].

$$N_{\text{predicted}} = \left\lceil \frac{\text{Predicted Control Activity}}{0.8 \times \text{Capacity of AMF}} \right\rceil . \quad (1)$$

Once the $N_{\text{predicted}}$ is defined, the Scaling Module adjusts the number of AMF instances according to the Algorithm 1. New AMF instances are created within the time interval in which the prediction is received (since the time to create an instance is usually less than 10 minutes, in practice). However, to mitigate the effect of oscillations, instances removal is made gradually.

Each instance has a "life" indicator that turns the instance off when the indicator reaches 0. At each time interval, the life of all instances is reduced, and three conditions are evaluated. The first case occurs when the number of current AMF instances ($\text{Total}_{\text{instances}}$) is lower than the required number ($N_{\text{predicted}}$). In this case, the life of all instances is increased, and new instances are created to meet the demand. When

[3]https://simpy.readthedocs.io/en/latest/

Algorithm 1: Active instance management algorithm

```
Input  : $N_{\text{predicted}}$: Predicted number of required instances;
         $\text{Total}_{\text{instances}}$: Total number of active instances;
         $\text{Life}_{\text{instance}}(i)$: Current lifetime of instance $i$;
Output: Updated states and $\text{Life}_{\text{instance}}(i)$;
if $Total_{instances} > 1$ then
    Decrease the lifetime of all instances;
end
if $Total_{instances} < N_{predicted}$ then
    Increase the lifetime of all instances;
    Increase the number of instances;
else if $Total_{instances} == N_{predicted}$ then
    Increase the lifetime of all instances;
else if $Total_{instances} > N_{predicted}$ then
    Increase the lifetime of the first $(\text{Total}_{\text{instances}} - N_{\text{predicted}})$
      instances;
foreach $instance(i)$ do
    if $Life_{instance}(i) == 0$ then
        Shut down instance $i$;
    end
end
```

$\text{Total}_{\text{instances}}$ and $N_{\text{predicted}}$) are equal, all instances have their lives restored so that they remain active. In the third case, only some instances have their lives restored. The last loop marks the instance for shutdown. Only in this case will the Load Balancer gradually act to reduce traffic to that instance.

The performance of the models in simulations was evaluated using three metrics: *Full Occupancy Count* (FOC), measures the number of events in which requests exceeded the maximum capacity of the AMF instances; *Number of Lost Requests* (NLR), the total number of requests not served throughout the simulation; and *Difference between Ideal and Predicted AMF Instances* (DIPI), which computes the error between the appropriate number of AMF instances to handle the current demand and the $N_{\text{predicted}}$ value, with negative values indicating undersizing and positive values indicating oversizing.

IV. Results

This section evaluates the models from three complementary perspectives: forecasting quality, computing efficiency (Section IV-A), and high-level operational performance (Section IV-B). Due to space limitations, we present results only for the five models with the best average MAE performance. The complete data, including the evaluation of all tested models, is available at our public repository on GitHub [4].

A. Forecasting and Computing efficiency

Table I presents the MAE for the five best-performing models in each cluster. Among them, BLR obtained the best overall performance, with an average MAE of 284.04, followed by the EWAR and SRPR models. These models demonstrated consistent behavior across clusters, especially in clusters 0 to 2. However, in scenarios with severe concept drift, a significant increase in errors is observed for all models, the most critical case being that of KNNR, which exceeded a MAE of 1200 in cluster 4.

Table II presents the average IT and LT for the models in Clusters 0 and 4 and the average across all clusters. The

[4]https://github.com/lasseufpa/5g-online-scaling-exp

TABLE I: MAE values in each cluster for the top-5 models.

Model	C0	C1	C2	C3	C4	Mean
BLR	129.39	142.45	148.21	425.98	574.17	**284.04**
EWAR	178.11	194.01	205.16	625.11	888.44	**418.17**
SRPR	218.68	245.98	255.10	732.09	1076.15	**505.60**
SKL2RR	205.04	232.08	262.99	760.83	1096.74	**511.54**
KNNR	263.39	293.09	304.94	969.41	1250.03	**616.17**

BLR model was the most efficient in IT with only 0.0063 ms, representing the most responsive model. Following, EWAR and SKL2RR showed the better the LT aspects, less than 0.25 ms, making them the lightest models. In contrast, the SRPR model presented a longer LT, around 13 milliseconds. This result reveals that although the model has a greater capacity to adapt to varied scenarios, it comes with a high computing cost. On the other hand, its IT is quite competitive compared to other models, being around 0.0838 ms.

TABLE II: IT and LT (in milliseconds) for the top-5 models in Clusters 0, 4, and their average.

Model	Metric	Cluster 0 (ms)	Cluster 4 (ms)	Mean (ms)
BLR	IT	0.0063	0.0063	**0.0063**
	LT	0.5164	0.5200	**0.5182**
EWAR	IT	0.0233	0.024	**0.0236**
	LT	0.1419	0.1383	**0.1408**
SRPR	IT	0.0830	0.0845	**0.0838**
	LT	13.0987	13.2574	**13.1781**
SKL2RR	IT	0.1092	0.1082	**0.1087**
	LT	0.2419	0.2400	**0.2409**
KNNR	IT	1.5907	1.5655	**1.5781**
	LT	2.3475	2.2323	**2.2899**

Excepting SRPR, the KNNR and SKL2RR models presented the highest IT and LT among the models. The performance of KNNR, in particular, is due to the way it operates, performing several distance calculations of the tested instance to the multiple past data. This approach implies high computational complexity, especially in continuous data streams.

Regarding the final model size, as shown in Table III, the EWAR and SKL2RR models remained extremely compact, using approximately 12.96 KB and 4.73 KB each, while SRPR occupied, on average, about 851 KB, which is more than 170 times larger than SKL2RR. Thus, one can point out that BLR maintains a balance between performance and computing costs. With a very low IT (0.0063 ms), an LT of less than 1 ms, and a fixed size of 57.41 KB, the model presents a good tradeoff for applications that require accuracy and stability without compromising computational resources.

TABLE III: Model size (in KB) for the top-5 models in Clusters 0 to 4 and their average.

Model	C0	C1	C2	C3	C4	Mean
BLR	57.41	57.41	57.41	57.41	57.41	**57.41**
EWAR	12.96	12.96	12.96	12.96	12.96	**12.96**
SRPR	909.77	800.56	932.59	914.28	698.22	**851.08**
SKL2RR	4.73	4.73	4.73	4.73	4.73	**4.73**
KNNR	17.01	17.01	17.01	17.01	17.01	**17.01**

B. Simulation Results

Table IV presents each model's NLR and FOC events across all clusters. The table shows that BLR, EWAR, and SRPR, the less error-prone models, are also efficient according to the high-level operational indicators. The BLR model obtained a reduced number of FOC events and relatively few NLR in total compared to others, especially in cases where the concept drift is more prominent. The KNNR model, which presented the worst average performance in terms of MAE, also suffered the most in terms of high-level indicators. This model recorded more than 100,000 NLR in cluster 4 and the highest number of FOC events in almost all clusters.

TABLE IV: NLR and FOC events for each model in all clusters.

Model	Metric	C0	C1	C2	C3	C4
BLR	NLR	1251	24	0	28584	43308
	FOC	2	1	0	2	7
EWAR	NLR	1251	24	0	31541	45758
	FOC	2	1	0	3	9
SRPR	NLR	1251	24	2886	28584	42728
	FOC	2	1	1	2	6
SKL2RR	NLR	1251	24	2886	31419	95342
	FOC	2	1	1	4	29
KNNR	NLR	2018	24	2886	70677	105032
	FOC	3	1	1	7	24

It is worth noting that the SKL2RR model, however, demonstrated similar performance to SRPR in error metrics and presented a significantly higher number of total occupancy events in Cluster 4, 29 events in total. This result is related to the model's tendency to project load growth linearly, even in the face of peaks, which contributes to the recurring underestimation of demand. This feature stands out even more when comparing the SRPR model with the BLR model, so that, despite being less precise on average, it was able to anticipate the system's needs more effectively, managing to obtain 580 fewer NLR than the BLR model.

Table V details the DIPI distribution in Cluster 4, revealing interesting patterns. In general, all models choose the correct number of AMF instances (DIPI = 0) most of the time. However, as the MAE increases, the dispersion of DIPI also increases. Thus, whereas the BLR, EWAR, and SRPR models maintain most of their errors close to zero, indicating more balanced predictions even in adverse scenarios, KNNR has a significant amount of moderate negative errors (DIPI = -1 to -2) in addition to a non-negligible amount of overestimates (DIPI > 0), which reinforces its unstable behavior.

TABLE V: DIPI distribution for each model in Cluster 4.

Model\DIPI	-4	-3	-2	-1	0	1	2	3	4
BLR	1	0	1	99	3970	71	0	1	0
EWAR	1	0	2	132	3870	137	1	0	0
SRPR	1	0	1	216	3777	147	0	0	1
SKL2RR	1	1	1	225	3763	152	0	0	0
KNNR	2	0	7	154	3737	192	49	2	0

V. Conclusion

This work evaluated the use of different *online* learning techniques for proactive scaling of 5G core network functions in the presence of concept drift. A *trace*-based simulator was built and fed with real user activity data to compare the behavior of an *online* prediction model against an *offline* model when faced with changes in traffic patterns. The results demonstrated that evaluating the performance of models exclusively based on the MAE is not enough to ensure an appropriate choice in real operational scenarios. For example, the SRPR model presented a MAE superior to the SKL2RR model, but obtained superior performance in scenarios with significant variations in traffic patterns. In these scenarios, reacting quickly and anticipating abrupt changes are essential to avoid losses.

As future work, we plan to implement a proactive scalability system and experiment realistic scenarios using the open source OpenAirInterface [5] 5GC software, evaluating how online learning models perform.

Aknowledgements

The Innovation Center partially financed this work, Ericsson Telecomunicações S.A., Brazil; and Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Finance Code 001.

References

[1] R. Li, B. Decocq, A. Barros, Y.-P. Fang, and Z. Zeng, "Estimating 5G network service resilience against short timescale traffic variation," *IEEE Transactions on Network and Service Management*, vol. 20, no. 3, pp. 2230–2243, 2023.

[2] M. P. Kuranage, E. Hanser, L. Nuaymi, A. Bouabdallah, P. Bertin, and A. Al-Dulaimi, "Ai-assisted proactive scaling solution for CNFs deployed in kubernetes," in *2023 IEEE 12th International Conference on Cloud Networking*, 2023, pp. 265–273.

[3] T. da Silva, T. Batista, F. Delicato, and P. Pires, "An online ensemble method for auto-scaling NFV-based applications in the edge," *Cluster Computing*, pp. 1–25, 2024.

[4] I. Alawe, A. Ksentini, Y. Hadjadj-Aoul, and P. Bertin, "Improving traffic forecasting for 5G core network scalability: A machine learning approach," *IEEE Network*, vol. 32, no. 6, pp. 42–49, 2018.

[5] M. P. J. Kuranage, L. Nuaymi, A. Bouabdallah, T. Ferrandiz, and P. Bertin, "Deep learning based resource forecasting for 5G core network scaling in kubernetes environment," in *8th IEEE International Conference on Network Softwarization*, 2022.

[6] V. Passas, N. Makris, Y. Wang, A. Apostolaras, A. Mpatziakas, A. Drosou, T. Korakis, and D. Tzovaras, "Artificial intelligence for network function autoscaling in a cloud-native 5G network," *Computers and Electrical Engineering*, vol. 103, p. 108327, 2022.

[7] I. Alawe, Y. Hadjadj-Aoul, A. Ksentinit, P. Bertin, C. Viho, and D. Darche, "An efficient and lightweight load forecasting for proactive scaling in 5G mobile networks," in *IEEE Conference on Standards for Communications and Networking*, 2018.

[8] A. Costa, R. Giusti, and E. M. d. Santos, "Analysis of descriptors of concept drift and their impacts," in *Informatics*, vol. 12, no. 1. Multidisciplinary Digital Publishing Institute, 2025, p. 13.

[9] A. Ferreira, K. Tavares, D. Vidal, S. Lins, A. Klautau, C. Bonato, and G. Gonçalves, "Online approach for proactive scaling of mobile core network functions," in *Anais do V Workshop de Redes 6G*. Porto Alegre, RS, Brasil: SBC, 2025, pp. 17–24.

[10] G. Barlacchi, M. De Nadai, R. Larcher, A. Casella, C. Chitic, G. Torrisi, F. Antonelli, A. Vespignani, A. Pentland, and B. Lepri, "A multi-source dataset of urban life in the city of milan and the province of trentino," *Scientific data*, vol. 2, no. 1, pp. 1–15, 2015.

[11] Q. Zhu and L. Sun, "Big data driven anomaly detection for cellular networks," *IEEE Access*, vol. 8, pp. 31 398–31 408, 2020.

[12] J. M. DeAlmeida, C. F. T. Pontes, L. A. DaSilva, C. B. Both, J. J. C. Gondim, C. G. Ralha, and M. A. Marotta, "Abnormal behavior detection based on traffic pattern categorization in mobile networks," *IEEE Transactions on Network and Service Management*, vol. 18, no. 4, pp. 4213–4224, 2021.

[13] R. Jain, *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling*, 1st ed. Wiley, 1991.

[14] J. Meng, J. Huang, Y. C. Hu, Y. Koral, X. Lin, M. Shahbaz, and A. Sharma, "Modeling and generating control-plane traffic for cellular networks," in *Proceedings of the 2023 ACM on Internet Measurement Conference*, ser. IMC '23. New York, NY, USA: Association for Computing Machinery, 2023, p. 660–677.

[5]https://openairinterface.org/