Pervasive and Mobile Computing 116 (2026) 102158

Contents lists available at ScienceDirect

Pervasive and Mobile Computing

journal homepage: www.elsevier.com/locate/pmc

Traffic analysis and resource adaptation in large-scale 5G multi-layer edge networks

Marcello Pietri [a,*], Natalia Selini Hadjidimitriou [a], Marco Mamei [a], Marco Picone [a], Enrico Rossini [a], Edoardo Maria Sanna [b], Jovanka Adzic [b], Andrea Buldorini [b]

[a] Department of Science Methods for Engineering, University of Modena and Reggio Emilia, Reggio Emilia, Italy
[b] Telecom Italia - TIM, Torino, Italy

ARTICLE INFO

Dataset link: NetMob 2023 Data Challenge (Original data)

Keywords:
Network optimization
5G mobile network ecosystem
Network virtualization
Mobile network data

ABSTRACT

In this research, we propose automating network management through data-driven intelligence, with a particular focus on anomalies and network traffic during specific events or periods. We analyze a large dataset collected by Orange mobile network operator in France with the goal of forecasting mobile demand for different classes of services. To model the underlying network infrastructure, we introduce a model for the underlying network based on a hierarchy of virtualization layers and slices. Building on this model, we propose algorithms to optimize the resources allocated to network slices and traffic distribution within the operator's network. Network performance is evaluated as the fraction of time the mobile traffic is within the capacity of the network. Our results demonstrate that dynamic reallocation of resources among slices, and dynamic load balancing (traffic shaping) between nodes notably improves network performance. These results provide insights into critical aspects related to future 5G network management.

1. Introduction

The advent of technologies such as Internet of Things (IoT), mobile robots, autonomous driving, mobile gaming, virtual and augmented reality will create an enormous pressure on the mobile networks. The next generation of mobile networks – 5G and beyond – will need to support services and applications with a wide range of different requirements in terms of Quality of Service (QoS). This is well summarized in recent ITU specifications that separate: Enhanced Mobile Broadband (eMBB), Ultra Reliable Low Latency Communications (URLLC) and Massive Machine Type Communication (mMTC) services as macro categories [1]. In this context, telecom operators will need to enforce ever stringent SLA (Service Level Agreement) with ever stringent budget constraints both in terms of hardware to be deployed (CAPEX) and operation costs (OPEX).

The key technology to meet such requirements is the combination of:

- data-driven solutions and machine learning models to forecast mobile demand and throughput;
- the virtualization of networks functions and servers to provide operators with unprecedented flexibility on how to allocate resources, re-route traffic and slice the network dynamically.

Overall, this combination allows operators to maximize the efficiency of the network by dynamically configuring and adapting deployed resources to actual and forecasted mobile demand for different classes of services [2].

* Corresponding author.
E-mail address: marcello.pietri@unimore.it (M. Pietri).

https://doi.org/10.1016/j.pmcj.2025.102158
Received 27 October 2025; Accepted 20 December 2025
Available online 27 December 2025

In this research, we propose automating network management through data-driven intelligence, with a particular focus on anomalies and network traffic during specific events or periods (e.g., gatherings, sporting events, concerts, etc.). We analyze the NetMob23 dataset [3] with the goal of forecasting mobile demand for different classes of services, and we provide algorithms to optimize the resources allocated to network slices and optimize traffic distribution (load balancing) within the operator's network to match the demand.

The NetMob23 dataset provides rich data about the spatio-temporal consumption of mobile applications in a developed country. This kind of data effectively represents the kind of data that is practically accessible from mobile operators with privacy-preserving limitations. Therefore, we think that the analysis and approach we develop in this work are compatible and applicable to future large-scale data available in this domain.

Building upon existing work by [4–6], the contribution of this paper is to provide a "full-stack" solution dealing with:

- Modeling the mobile network (Section 2). We create a network model based on vertical partitioning using network slices and horizontal partitioning across multiple layers of core and edge computing nodes. Although our model is mainly data-driven, the main drawback is that we had to make a number of assumptions on the network architecture and operation to fill-in missing information. Nevertheless, the proposed methods and algorithms can be tuned to specific deployment once information are available.
- Demand forecasting (Section 3). We trained a novel deep-learning model – based on LSTM – to forecast network demand for different classes of services. We focus on predicting spikes that can saturate network resources and capacity.
- Network optimization algorithms (Section 4). We developed high-level optimization algorithms to improve network performance by sharing/re-balancing resource among slices, and sharing/re-balancing traffic among network nodes.

We extensively evaluate all the models and algorithms on the basis of real-world data collected in the city of Nantes in France [3]. All experiments and results are described in Section 5. Section 6 surveys related works and, finally, Section 7 concludes.

2. Network modeling and slicing

One of the main strategies behind 5G networks revolves around leveraging network virtualization, a process that transforms the traditional rigid hardware structure into a dynamic cloud-based architecture. Thanks to virtualization, hardware-dependent network functions, such as baseband processing, mobility management, load balancers, evolved packet core (EPC) service, etc. are implemented by means of software-based Virtual Network Functions (VNFs) operating within a versatile telco-cloud environment. Application servers can also be run in such telco-cloud environment, so as to improve their performance (e.g., by reducing latency). In particular, network deployment leverages this newfound flexibility, relying on two main patterns:

1. Virtualization facilitates the establishment and comprehensive utilization of a hierarchical network structure featuring multiple edge layers (e.g., Multi-access Edge Computing - MEC servers [7]) with each layer dedicated to distinct network and application functions. Each layer is composed by a number of servers/data centers processing data from a subset of antennas or intermediate edge nodes. On the Radio Access Network (RAN) layer the antennas are typically associated with the Base Station (BS), which can be referred to as either an eNodeB (evolved Node B) or a gNodeB (next-generation Node B) in Long-Term Evolution (LTE) networks or a BTS (Base Transceiver Station) in older 2G-3G cellular network generations. *In general, one critical tasks of this flexible architecture is how to dynamically route and balance traffic between nodes of multiple layers [4].*
2. On the other hand, by embracing network virtualization, it becomes possible to establish multiple virtual iterations of the entire network, referred to as network slices. This arrangement ensures that the Key Performance Indicators (KPIs) and QoS demands of current and future mobile applications are met effectively. These slices can be effortlessly tailored by fine-tuning the functionality and positioning of VNFs. This approach generates a series of logical networks that overlay the physical infrastructure. Each of these logical networks is meticulously designed to accommodate precisely defined SLAs, aligning with the distinctive requirements of various service providers. *In general one critical tasks in this context is how to allocate resources to multiple slices [5,6].*

We propose modeling the mobile network according to the above two patterns.

The mobile network consists of a set of network nodes/data-centers comprising VNFs and application servers (see Fig. 1). Nodes are organized in hierarchical layers. At the bottom there are the *edge* layers. Such network nodes are associated to a subset of BS and are in charge of managing traffic to/from such BS. Top layers represent the *core* network, such network nodes are in charge of processing data from lower layers and interacting with cloud services and resources outside the mobile operator network. Network traffic runs from BS to edge nodes, to core nodes and vice versa.

Network nodes in all the layers should be dimensioned to meet the expected traffic demand. We consider a simplified network model, where data and traffic are directly associated with resource consumption without accounting at the moment possible non-linear relationships between traffic and resources. We assume that the mobile traffic generated by BS is divided into different classes of services. The resources in each network node are divided between multiple network slices. There is a network slice for each class of service, and each slice is in charge of processing the traffic of that class of service (for example, the slice associated to video-services will handle all the traffic generated by videos). It should be noted that in this analysis, we have primarily focused on higher-level algorithmic and modeling issues, deliberately setting aside considerations of specific network functions capacities,

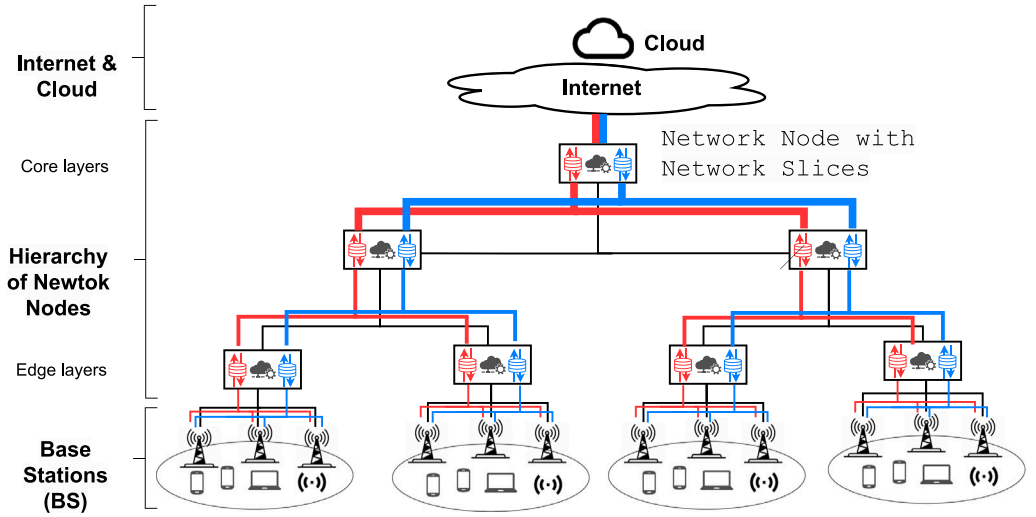

Fig. 1. A schematic representation of the hierarchical and sliced architecture associated with the 5G edge networks. The network is partitioned vertically on the basis on network slices (red and blue nodes and traffic fluxes), and partitioned horizontally on the basis of multiple layers of core and edge computing nodes.

such as distinctions between RAN or 5G core data plane versus user plane functions, due to the unavailability of detailed data and the specific scope of our study. We posit that this architecture is general enough to accommodate a wide range of modern network deployments. Despite the abstract nature of our model, which does not delve into specific network functions and capacities, it provides a robust framework for analyzing traffic and resource distribution across different network slices and hierarchical layers. This allows for scalable simulations and theoretical assessments which are essential in the planning and development of efficient mobile networks.

Finally, in order to link the traffic demand on a node with a measure of performance we introduce, for each node n and slice s, a capacity $C_{n,s}$ representing the amount of traffic that the resources associated to that node in the slice can handle before network saturation. This is a strong simplification as $C_{n,s}$ subsumes different resources in terms of: networking, memory, storage and computation. The capacity should effectively handle normal traffic but it might be saturated during peak events.

Referring to the traffic in a node n in slice s at time t as $T_{n,s,t}$, we assume that the network performance degrades if $T_{n,s,t} > C_{n,s}$. In order to optimize the network operations we will try to minimize the time intervals in which $T_{n,s,t} > C_{n,s}$ and the amount of the excess $T_{n,s,t} - C_{n,s}$.

Of course, the number of layers, the number of nodes in each layer, how many BS are managed by each node, the number of slices and class of service and the capacity $C_{n,s}$ values are critical information entirely depended on the actual network deployment. In Section 5, we provide a data-driven hypothesis of the network on which to base our experiments. However, there could be also other approaches on which to instantiate the model (e.g., obtain the network structure directly from the network operator).

3. Traffic forecasting

Dynamic resource allocation, particularly in scenarios of network stress, necessitates the ability to anticipate and adapt to fluctuating service demands. For efficient resource orchestration, network operators must possess the capability to re-assign resources dynamically in advance to a predicted surge in demand. Accurate forecasting is critical: underestimating future traffic might lead to resource under-provisioning and network saturation that may result in SLA violations, undermining the network's reliability. Overestimating future traffic might lead to excessive resource provisioning that can lead to economic inefficiency.

We applied a time-series forecasting algorithm, based on Long Short-Term Memory (LSTM) [8], to forecast future traffic on each node and slice of the network. This network models time interrelations by feeding or blocking past information to predict the future. Then, we will balance resources and traffic accordingly to meet the predicted demand. In this context, traffic forecasting is inherently linked to the capacity of the network to timely reconfigure its resources. For example, forecasting traffic T minutes ahead is useful if the network can be reconfigured at least every T minutes. This creates a challenging trade-off between aggregating traffic over large time windows (e.g., 1 h), which typically results in smoother time series that are easier to forecast but limits the frequency of network reconfiguration (e.g., once every hour), versus aggregating traffic over shorter periods (e.g., a few minutes), which generates more erratic time series that are harder to forecast but allows for more frequent network reconfiguration. With regard to this point, it is important to consider that dynamic resizing also introduces several costs. At the technical level, scaling requires significant control-plane signaling across RAN, transport, and core domains, adding overhead and increasing orchestration complexity. The reallocation of resources can cause transient service disruptions, such as brief latency spikes, jitter, or packet loss,

which are particularly critical for URLLC. Furthermore, scaling operations often involve spinning up or decommissioning virtualized network functions, leading to additional computational and energy costs.

In practice, modern cloud-native implementations complete resizing actions within a few tens of seconds, making it feasible to perform adjustments at relatively coarse intervals, such as once per hour, for URLLC applications, without significant performance degradation. Of course, there is also an absolute upper bound on the frequency of network reconfiguration, defined by technological and operational limits in managing the network [9].

These factors highlight the need for efficient traffic forecasting and predictive scaling mechanism to balance performance and cost.

In Section 5.3 we present results of our forecasting approach. In our experiments, we set forecasting and reconfigurations 1 h ahead focusing on URLLC scenario, but our approach is general and can be also tuned to coarser (e.g., daily) forecasting and reconfigurations.

Forecast traffic demand is the basis to run the optimization described in the following section.

4. Network optimization

Given the above scenario, we devised a methodology to optimize network operations on the basis of forecast traffic on the different slices.

Our base hypothesis is that the network allocates resources fairly statically according to the modeled capacities $C_{n,s}$. If traffic is manageable $T_{n,s,t} < C_{n,s}$ the network does not modify its resources. Vice versa, if the traffic is predicted to grow beyond capacity, the network can temporarily adjust resources to meet the surge in demand. Once traffic is predicted to diminish, the network will return to its base allocation $C_{n,s}$.

Our approach is based on two complementary mechanisms:

1. **Resource sharing/re-balancing among slices.** Our network model is based on a set of hierarchical nodes/data-centers processing network traffic. The resources in each node C are divided to handle multiple slices. C_s are the resources allocated to handle slice s ($C = \sum C_s$). If the traffic associated to a slice $s1$ at time t is greater than the capacity $T_{s1,t} > C_{s1}$, the network under-performs. If there is another slice $s2$ in the same data-center with spare capacity $T_{s2,t} < C_{s2}$, some resources could be reallocated from $s2$ to $s1$. The idea of this mechanism is to apply this pattern on the basis of forecast traffic: if the network forecasts that at future time t, $T_{s1,t} > C_{s1}$ and $T_{s2,t} < C_{s2}$, then it re-balances resources in advance, leaving the total amount C unchanged. The Algorithm 1 formalizes this concept: slices *in need* of resources borrows them from slices *in surplus* of resources. In particular, in our algorithms, each slice shares equally its surplus to slices in need. In our model, this translates in a temporary change of C_s values.
2. **Traffic sharing/re-balancing among network nodes/data-centers.** The dual approach is to re-balance traffic instead of resources: if the above resource-balancing-within-a-node cannot accommodate all the demand ($\sum_s T_{s,t} > C$), a node can re-route part of the traffic to other nodes. This re-route can take place: *(i)* among the nodes of the same network layer, e.g. part of the traffic to a given edge node is routed to a nearby node at the same edge layer (i.e., horizontal offloading). *(ii)* among the nodes at different network layers, e.g. part of the traffic to a given edge node is routed to and processed by a node at an upper layer of the hierarchy - a core node (i.e., vertical offloading). The Algorithm 2 formalizes this concept. The algorithm is very similar to Algorithm 1: nodes *in need* of resources lends traffic to nodes *in surplus* of resources.

These mechanisms, albeit abstract, are integral in optimizing resource and traffic patterns across the network, serving as a high-level representation of the network's complex operational demands to achieve resource rebalancing. While we acknowledge the potential for increased congestion and latency due to traffic sharing among network nodes, our study was concentrated on data centers, not on granular network modeling. This focus restricted our examination of how traffic distribution within a single node might affect service performance. Moreover, although our algorithms are crafted to distribute resources fairly across network slices, the absence of specific data prevented us from considering individual QoS requirements or revenue potentials. We plan to refine these algorithms by integrating weighted allocations that reflect these factors in future work, based on a more robust dataset. Despite these limitations, the preliminary results from these algorithms are instrumental in elucidating the potential benefits of network optimization under these constraints.

5. Experiments and results

In this section, we describe the experimental setup and results for each of the above modeling steps.

5.1. Dataset

In our experiments, we relied on the NetMob23 dataset [3], which consists of network traffic generated by customers of the Orange telecommunications and digital service provider. This dataset consists of the traffic demands (both downlink — DL and uplink — UL) generated by 68 popular mobile services over 20 metropolitan areas in France during 77 consecutive days (March-16 to May-31) in 2019. Data is geo-referenced at a high resolution of 100 m × 100 m tiles tessellating the area and with a 15 min time resolution. In the NetMob23 dataset, for the purpose of preserving sensitive information, all traffic data has been normalized using

Algorithm 1 Algorithm to share resources among slices in the same node / data center

For each network node:

$\forall t \in$ time steps
$\forall s \in$ network slices
$T_{s,t} \leftarrow$ network traffic of slice s at time t
$C_s \leftarrow$ capacity of slice s
$in\_need \leftarrow \{s : T_{s,t} > C_s\}$ ▷ slices that need more resources
$tot\_need \leftarrow \sum(T_{s,t} - C_s) \ \ \forall s \in in\_need$
$in\_surplus \leftarrow \{s : T_{s,t} < C_s\}$ ▷ slices that have a surplus of resources
$tot\_surplus \leftarrow \sum(C_s - T_{s,t}) \ \ \forall s \in in\_surplus$
$available \leftarrow min(tot\_need, tot\_surplus)$ ▷ resources that can be shared
if $available > 0$ **then**
  $to\_lend_s \leftarrow (C_s - T_{s,t}) * available/tot\_surplus \ \ \forall s \in in\_surplus$
  $to\_borrow_s \leftarrow (T_{s,t} - C_s) * available/tot\_need \ \ \forall s \in in\_need$
  $C_{s,t} \leftarrow C_s - to\_lend_s \ \ \forall s \in in\_surplus$ ▷ dynamic capacity for time t
  $C_{s,t} \leftarrow C_s + to\_borrow_s \ \ \forall s \in in\_need$ ▷ dynamic capacity for time t
end if

Algorithm 2 Algorithm to re-route traffic among nodes of the same or of different layers

For each couple of layers L_n, L_s (could be $L_n = L_s$) – L_n stands for layer that could be in *need*, L_s stands for layer that could be in *surplus*

$\forall t \in$ time steps
$T_{n,t} \leftarrow$ network traffic of node n at time t
$C_n \leftarrow$ capacity of node n
$\forall n \in$ node $\in L_n$
$in\_need \leftarrow \{n : T_{n,t} > C_n\}$ ▷ nodes that need more resources
$tot\_need \leftarrow \sum(T_{n,t} - C_n) \ \ \forall n \in in\_need$
$\forall n \in$ node $\in L_s$
$in\_surplus \leftarrow \{n : T_{n,t} < C_n\}$ ▷ nodes that have a surplus of resources
$tot\_surplus \leftarrow \sum(T_{n,t} - C_n) \ \ \forall n \in in\_surplus$
$available \leftarrow min(tot\_need, tot\_surplus)$ ▷ resources that can be shared
if $available > 0$ **then**
  $to\_borrow_n \leftarrow (C_n - T_{n,t}) * available/tot\_surplus \ \ \forall s \in in\_surplus$ ▷ traffic that can be borrowed by nodes in surplus
  $to\_lend_n \leftarrow (T_{n,t} - C_n) * available/tot\_need \ \ \forall s \in in\_need$ ▷ traffic that can be lent by nodes in need
  $T_{n,t} \leftarrow T_{n,t} + to\_borrow_n \ \ \forall n \in in\_surplus$ ▷ re-routed traffic for time t
  $T_{n,t} \leftarrow T_{n,t} - to\_lend_n \ \ \forall n \in in\_need$ ▷ re-routed traffic for time t
end if

a uniform random value. Consequently, traffic is not associated with a specific unit (such as bytes); nevertheless, it remains entirely comparable across different spatial and temporal contexts [3]. Therefore in our analysis units of measurement for traffic will not be employed.

Our method is direction-agnostic and can be applied separately to UL and DL. In this study, we aggregated UL and DL to streamline the exposition. We acknowledge that this is a strong simplification, but since network architecture details are missing (see Section 5.2), this aggregation has a negligible impact on our methods and results.

This kind of data is fully compatible with the General Data Protection Regulation (GDPR) and is a relevant benchmark for real-world dataset that can be provided by mobile phone operators. We think that the limitations present in this dataset (due to privacy constraints) are representative of future large-scale dataset to be shared in this domain.

In this work, we focus on the city of Nantes. The city is the sixth largest in France, with a population of about 320,000 in Nantes proper, and a metropolitan area of nearly 1 million inhabitants. Despite this focus, the approach presented and experiments are fully applicable also to all the other cities in the NetMob23 dataset. Additionally, we obtained data from the Observatoire 2G, 3G, 4G, 5G dataset [10] on the location of the circa 200 BS in the city of Nantes (including eNodeB, gNodeB or BTS).

Finally, we collected data about large events and happenings taking place in Nantes during the monitored period as they can significantly impact network demand. In particular, other than public holidays, we utilized Google Trends to retrieve events only if they were published prior to the date of the event in question. We focused on three distinct geographical groups: France, Pays de la Loire, and specifically Nantes. Within each group, our analysis covered five events categories, including general trends, sports, and news with events. Examples include "Nantes, News, International Fair" for the Nantes-specific group, and for France, categories such as sports with events like "Champions League". The data, weighted for geographical relevance, was used to create a dataset with values normalized between 0 and 1 for each date depending on events being held.

5.2. Network modeling and slicing

In this section we describe how we instantiated the model presented in Section 2 in our experiments.

5.2.1. Network model and hierarchy

In order to establish a connection between NetMob23 data and the network's operation, considering the limited knowledge of the mobile network architecture in the dataset, beyond the radio access, we rely on assumptions drawn from the characteristics of real-world networks in similar geographic contexts. These assumptions bridge the gap between collected data and network functionality, enabling valuable insights and effective network management strategies.

As a preliminary step, we map the NetMob data – where mobile traffic is aggregated by geographic tiles – to individual BS. For this purpose, we adopt a straightforward approach: each tile's traffic is assigned to its nearest base station. We did not investigate more advanced algorithms for this matching as the original data collected by the Orange network already associated traffic to base stations. Here, we are basically undoing some of the operations undertaken to prepare the NetMob23 dataset. Therefore, in principle, this kind of matching could be easily provided by the operator.

As a second step, we need to instantiate the hierarchy of network nodes described in Fig. 1. Drawing from our knowledge of national infrastructure, it is plausible to envision a structured network configuration for a mid-sized city (100–500 K inhabitants) with approximately four hierarchical layers:

- Level 0: Site/Antennas - BS (210 BS in our data)
- Level 1: Far Edge/MEC, encompassing about 5 BS per cluster (40 clusters)
- Level 2: Near Edge, including 4 clusters (i.e. 50 BS)
- Level 3: Core Network covering the entire city, 1 node

Specifically, we applied an approach inspired by the one presented in [5,6]. We run a clustering algorithm on the BS using geographical distance as a key metric, and ensuring that clusters have about the same number of nodes. Using geographical distance ensures that BS in the same cluster are nearby, thus reducing the cost to connect (e.g., via optics fiber) the BS to the network node. Having the same number of nodes per cluster ensures that the overall traffic is similar in each cluster. We implemented the clustering as a simple modification to the standard K-means algorithm to ensure equal cluster size [11]. We associate a network node to each centroid resulting form the clustering. Therefore the far-edge layer will have 40 network nodes connected to about 5 BS and managing traffic from them. The near-edge layer will have 4 network nodes connected to 10 far-edge nodes, i.e. 50 BS. The core layer is comprised by a single network node/data center covering the entire city. Fig. 2 illustrates the results. The top row shows BS deployed across the city colored according to the associated cluster. Bottom row is the distribution of overall traffic in the different clusters.

As the aggregation level progresses towards the root node, the volume of traffic increases accordingly. At the root node or data center, the traffic from the entire city is consolidated. Also the traffic is fairly equally divided across clusters, as illustrated in Fig. 2 (bottom-left). However, when examining the average traffic per base station within each cluster (Fig. 2, bottom-right), we observe greater variability in peripheral areas. As the aggregation level rises, the traffic tends to become more periodic and regular, reflecting the characteristics of aggregated data.

The proposed architecture includes only the *minimal* set of specifications needed to enable our analysis. On one hand, this modeling choice is interesting because it provides a link from traffic data, such as those examined in this paper, to a hypothetical mobile network. On the other hand, the architecture is admittedly limited and under-specified. For example, our model does not differentiate between UL and DL, while 3GPP defines QoS parameters and KPIs per direction, so UL and DL may show different parameters and performance values. Nevertheless, lacking real data on the UL and DL network architecture, we disregarded these differences. This is also why we aggregated UL and DL traffic (as described above), as they are equivalent in our simplified network model.

5.2.2. Service clustering and network slices

In order to meet the QoS requirements of the multiple services accessed by mobile network users, telecom operators are likely to group services in multiple classes according to their requirements (e.g., in terms of latency and throughput) and assign to each class a network slice with resources properly optimized for that class of services. This typically ends-up creating classes/clusters like the eMBB, URLLC, mMTC as defined in [1].

As our dataset does not contain requirements' information for specific services, we had to design a mechanism to cluster services in classes according to their traffic usage patterns, so as to have services that are consumed in the same way/amount being clustered in the same network slice. This is a limitation of our work: there could be services with very different traffic patterns that have similar requirements and should be assigned to the same slice's resources. However, this is the best we could do with the data at hand.

More in detail, if $t_{s,d,h}$ is the traffic generated by service s on day d at hour h, for each service s, we can create the following features:

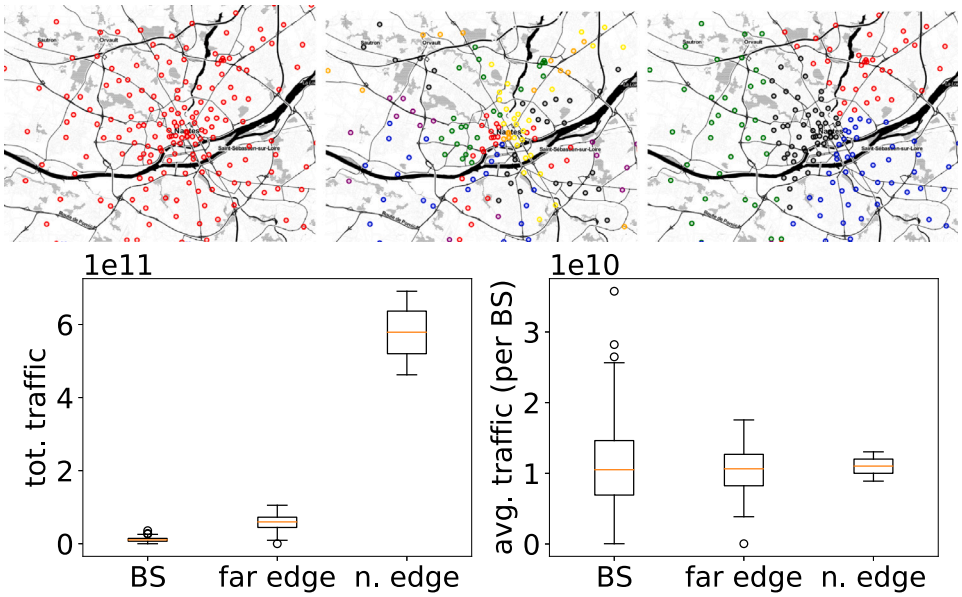

Fig. 2. (top-row) Antennas deployment in the city of Nantes, association of antennas to far edge layer (40 clusters), association of antennas to near edge layer (4 clusters). **(bottom)** nodes' traffic distribution at antenna, far edge and near edge layers: total traffic distribution and mean traffic distribution within the cluster.

- The normalized hourly average traffic $f_{s,h}$, that, for each hour, represents the fraction of daily usage of that service. This basically amounts at 24 features, for each service, computed as follows:

$$f_{s,h} = \frac{\sum_d^N t_{s,d,h}}{\sum_{d,h} t_{s,d,h}}$$

  They represent how a given service is consumed during the day.
- The fraction of the total traffic generated by a given service. This is computed as

$$tot_s = \frac{\sum_{d,h} t_{s,d,h}}{\sum_{s,d,h} t_{s,d,h}}$$

We then use standard k-means clustering on the resulting features. We empirically set N = 5 clusters. In general, in initial 5G deployments, operators typically implement three main network slices for eMBB, URLLC, and mMTC. As services expand to support advanced verticals like Industry 4.0 or connected vehicles, the number of slices can grow to tens of slices, balancing service diversity with operational complexity. Therefore, N = 5 represent a reasonable set up for our experiments.

Fig. 3 shows the results. The algorithm identifies a cluster with only one service, i.e., Instagram – that alone amounts at almost 20% of the traffic. Then other four clusters both related to overall traffic consumption and to their daily pattern. We assume that the network operator creates a network slice associated to each cluster (5 slices in our experiments). In general, it is worth noticing that with more data and requirements' specifications, more advanced clustering algorithms could be implemented [12].

5.2.3. Network nodes' capacity

The last part of our network modeling approach is to assign capacities to each node n and slice s. To set $C_{n,s}$, we applied the approach proposed in [5,6]: the operator engages to guarantee that the traffic demand of the slice is fully serviced during at least a fraction of time. The traffic is computed as the average over discrete-time intervals of a given size. So, for example, the operator set $C_{n,s}$ to meet the hourly demand on 80% of the cases.

Fig. 4 illustrates the approach. To compute the resulting $C_{n,s}$ it is possible to compute the Cumulative Distribution Function (CDF) of the traffic of slice s in node n, and find the value (i.e., $C_{n,s}$) associated to a given fraction of the cases.

As an upper bound for the analysis of $C_{n,s}$, we applied the empirical idea that the capacity should be set to manage the average of "difficult" circumstances. Computing the $C_{n,s}$ as the average daily peak (maximum) of traffic results in about 97% of traffic being lower than the capacity.

In general, we experimented with $C_{n,s}$ able to meet hourly demand from 70% to 95% of the cases. This will be the base capacity for each slice upon which we will instantiate our optimization approach.

Given the capacity, we will analyze the performance of the network in terms of how many times the demand for a given class of services overcomes its capacity and the amount of capacity deficit in that cases.

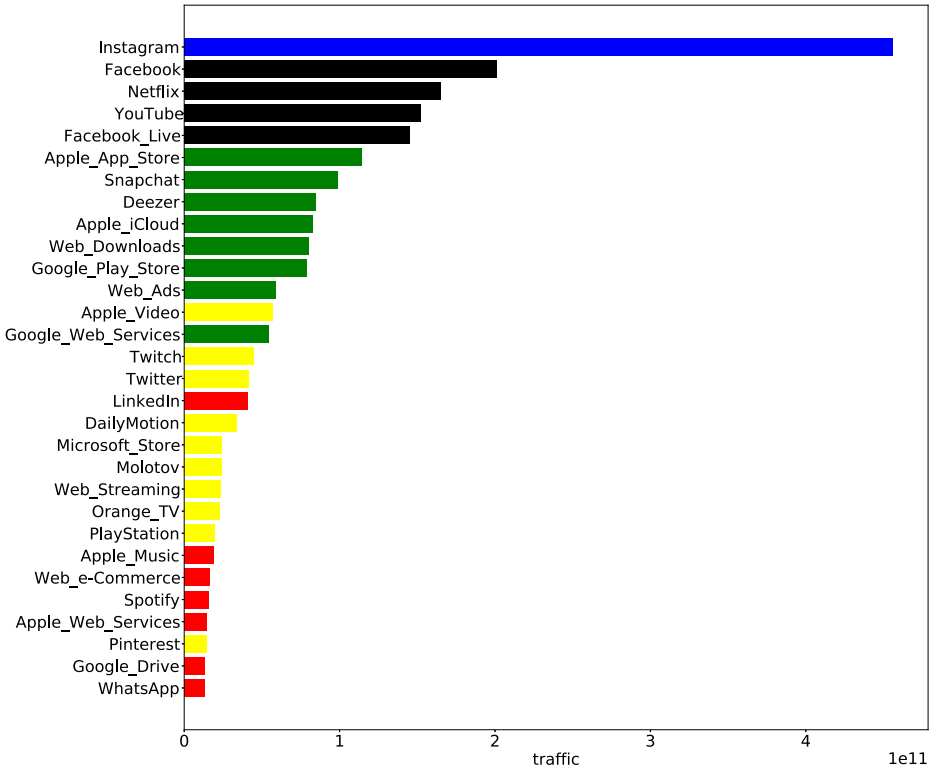

Fig. 3. Clustering of services in application classes associated to network slices. The color represents the cluster (only the top 30 services in terms of traffic are displayed).

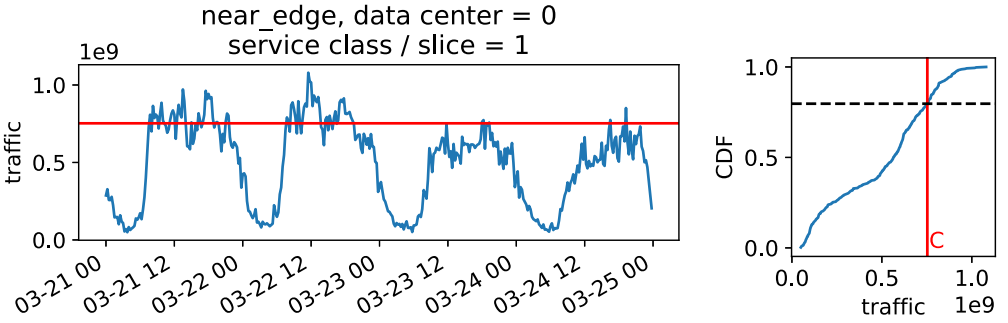

Fig. 4. Example of capacity C allocation for a given node and slice to meet the hourly demand on 80% of the cases. **(Left)** time series of the class of services being considered. **(Right)** CDF of the demand. In both cases the red line is the capacity to be set.

5.3. Traffic forecasting

We perform traffic forecasting on the five groups of services and each node using LSTM. We train the LSTM on the 80% of hourly data from 16 April 2019 to 16 May 2019 to predict the traffic in the next hour. In order to improve performance we introduced two important modifications to standard LSTM approach:

1. Following [13], we employ a customized direction-aware, piecewise loss function (Fig. 5) with an ε-insensitive band, a global cap K (maximum capacity), and a short linear shoulder up to $\varepsilon\alpha$ (nearly flat at α). Under-provisioning ($e_t < 0$) is penalized linearly, while over-provisioning ($e_t > \varepsilon\alpha$) grows smoothly and then saturates to K via a *Tukey-like (bisquare-inspired) smooth saturating* profile [14].
   Let d_t be the actual demand and p_t the provisioned capacity. Define the signed error $e_t = p_t - d_t$ and an $\varepsilon > 0$ insensitive band. Following [13], we encode the SLA–overprovisioning trade-off via $\alpha \in (0, 1)$ (ratio of SLA-violation vs. overprovisioning costs), and enforce a global cap $K > 0$. To keep a small positive cost right after the shoulder (instead of exactly zero), we use

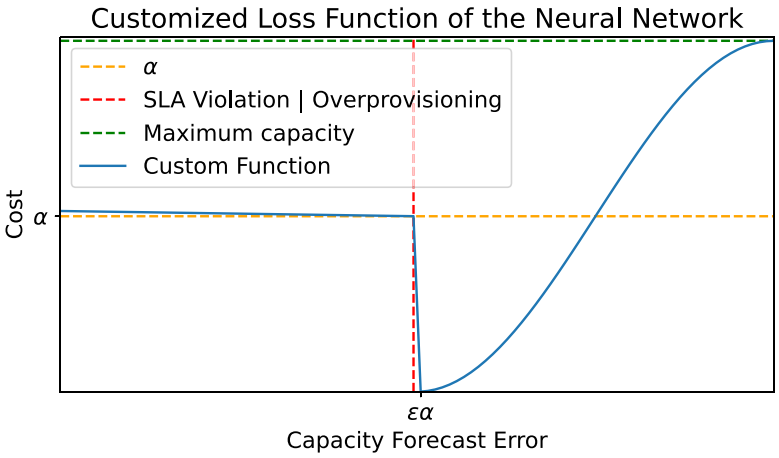

Fig. 5. Direction-aware provisioning loss with an ε-insensitive band, global cap K (maximum capacity), and a short linear shoulder up to $\varepsilon\alpha$ (nearly flat at α). Under-provisioning ($e_t < 0$) is penalized linearly and capped at K. For $0 < e_t \le \varepsilon\alpha + \delta$ the curve descends from α to a small positive level η; for $e_t > \varepsilon\alpha+\delta$ it follows a Tukey-like (bisquare-inspired) smooth saturating profile (raised-cosine/quintic smootherstep), increasing roughly quadratically at first and flattening to K over width Δ with zero slope at the plateau. The loss is continuous and bounded in $[0, K]$; kinks occur at $e_t = 0$ and $e_t = \varepsilon\alpha + \delta$.

$\delta > 0$ and $\eta \in (0, K)$ and set $e_{\text{end}} = \varepsilon\alpha + \delta$ and $m_{\text{sh}} = (\alpha - \eta)/e_{\text{end}}$. With a smooth saturating tail controlled by $\Delta > 0$ and

$$\sigma(t) = \tfrac{1}{2}\big[1 - \cos(\pi\ \text{clip}(t, 0, 1))\big], \quad \text{clip}(u, 0, 1) = \min\{1, \max\{0, u\}\},$$

the per-step loss is

$$\ell_t = \begin{cases} \min\{K,\ \alpha - \varepsilon\, e_t\}, & e_t \le 0, \\ \max\{\eta,\ \alpha - m_{\text{sh}}\, e_t\}, & 0 < e_t \le e_{\text{end}}, \\ \eta + (K - \eta)\,\sigma\big((e_t - e_{\text{end}})/\Delta\big), & e_t > e_{\text{end}}\ , \end{cases}$$

which is continuous and bounded in $[0, K]$, with kinks at $e_t = 0$ and $e_t = e_{\text{end}}$ (subgradient-friendly). If strict C^1 smoothness is required, the linear parts can be replaced by a pseudo-Huber term or the raised-cosine σ can be substituted with a quintic smootherstep, while preserving the economic meaning of α and the plateau [13,14].

2. We explicitly provide the neural network with features associated to events taking place in the area that were planned in advance (e.g., public holidays, sport events, festivals), as it is reasonable that the network operator can plan accordingly. Of course, this kind of events can have a notable impact on the network traffic demand [15–17]. From a technical perspective, the presence of public holidays, weekends, the day of the week, and the time of the day is represented with one-hot encoding variables. Events are weighted for geographical relevance (proximity to the cell) with a 0–1 continuous variable.

We test some hyper-parameters in different experiments using *Optuna* [18] and selected the best combination. We varied the number of epochs from 50 to 500, the number of neurons from 50 to 500, the batch size from 2 to 128, and the learning rate from 10^{-5} to 10^{-1} with a number of experiments set to 10^3 (*Optuna n_trials*). In the subsequent step, we deployed the same hyper-parameters for all clusters and far edge/MEC. We employed a hyperbolic tangent activation function and the output layer has a linear activation function and one neuron. Concerning the optimization algorithm, we deployed the RMSProp for the prediction of far edge/MEC traffic of cluster 3 and cluster 4. For the three clusters 1, 2 and 5, we deployed the Adam [19] optimization framework.

Fig. 6 shows the distribution of the R^2 in each cluster. Most of the far edge/MEC predictions have an R^2 included between 86% and 99%, with a median of 96%. There are some outliers due to the fact that it was not possible to predict some unexpected events. The resulting forecasting algorithm is very accurate.

It is worth reporting that in preliminary experiments, using standard time series forecasting, we encountered a significant challenge: our forecast tended to underestimate peak traffic. This underestimation is crucial to the efficacy of our optimization algorithm; if peak traffic is not accurately predicted, the system may allocate insufficient resources, despite their availability within other nodes and slices. The current approach, especially due to the custom loss function, effectively addressees the issue. Fig. 7 shows the prediction results for one far edge — number 25, of cluster 3 corresponding to the Instagram service. The network slightly overestimated the traffic (for instance on 30 May 2019) because of the peaks on 29 May 2019 due to the Europa League Final (Chelsea - Arsenal and other events in Nantes) leading to enhancements in performance.

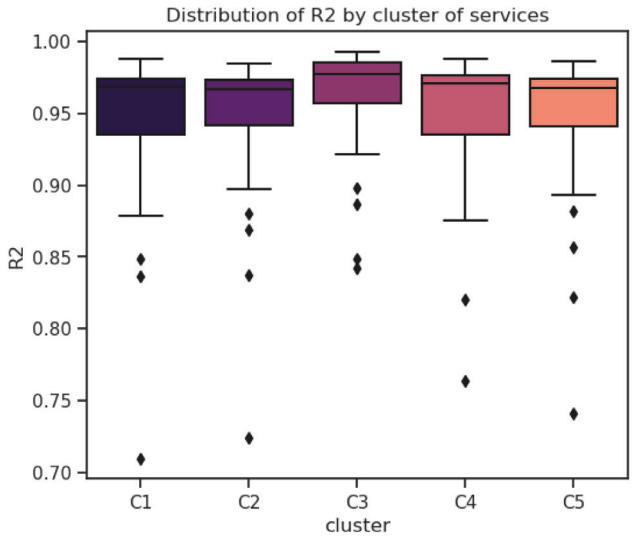

Fig. 6. Prediction results. The boxplots show the distribution of the R^2 in each cluster of services.

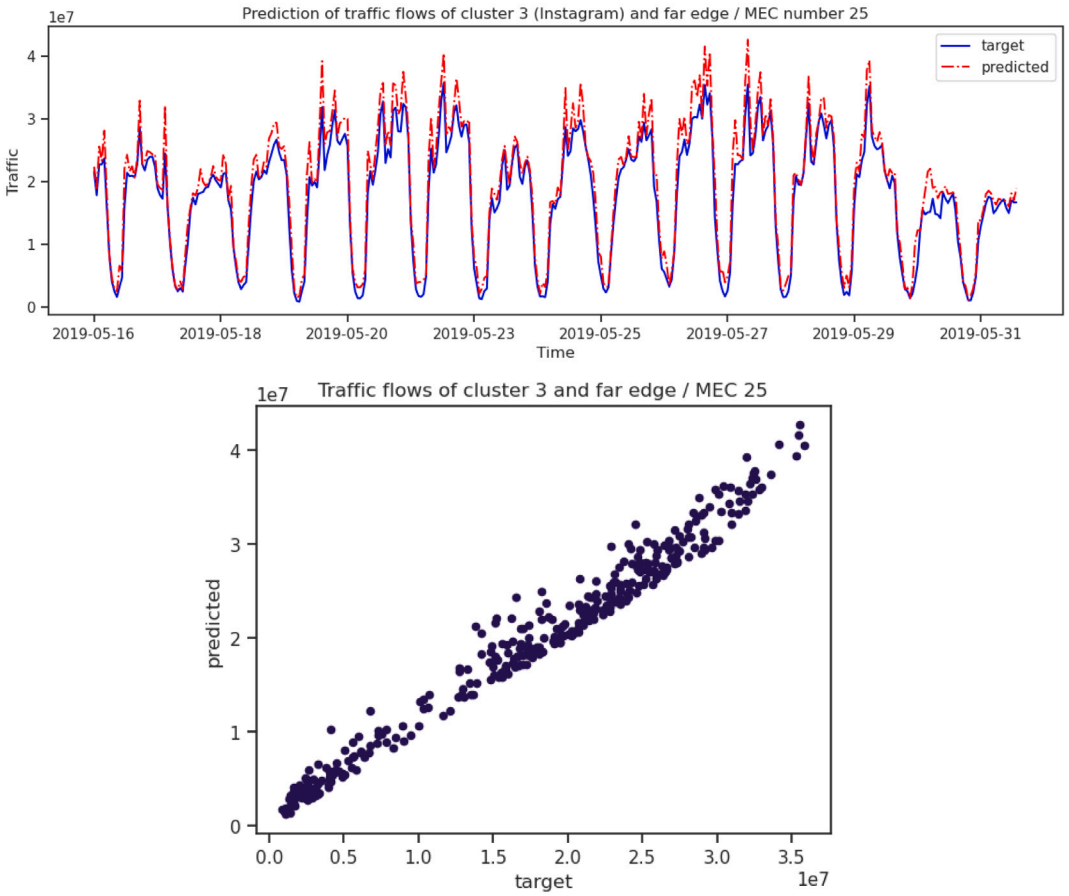

Fig. 7. Prediction results of far edge/MEC number 25 in cluster 3. The figure shows that the LSTM network predicts reasonably well the peaks of traffic. We performed the predictions on an hourly basis and used the results for dynamic capacity optimization.

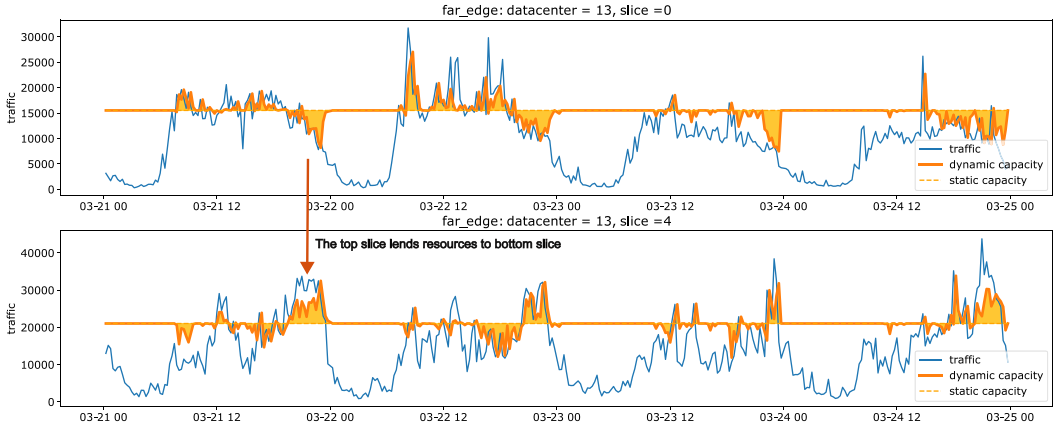

Fig. 8. Dynamic capacity optimization. The slice in the bottom graph has a peak demand at the beginning of the series (about 21/3/2019 10 pm). To meet such a peak, the capacity increases (orange line), with respect to the base capacity (dashed orange line), i.e., this slice borrows resources from other slices. Vice versa, the traffic in top graph is lower than its base capacity at the same time, this slice can lend resources to other slices. Overall, the total resources of the data center ($\sum_s C_s$) remains constant.

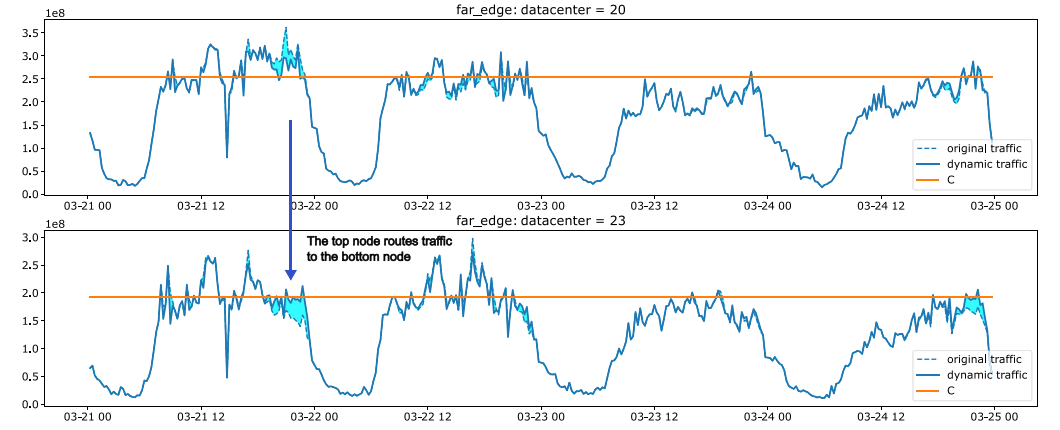

Fig. 9. Dynamic traffic optimization. The data center in the top graph as a traffic spike at about 21/3/2019 10 pm. This spike is well above the data center overall capacity ($\sum_s C_s$). The traffic can be re-routed to other nodes of the same layer, or even to nodes of other layers, granted that there are data centers that have spare capacity.

5.4. Network optimization

We run experiments to apply the proposed optimization algorithms to our case study. We present results of dynamic capacity optimization (see Algorithm 1) and dynamic traffic optimization (see Algorithm 2). We also show final results in combining the two approaches.

Fig. 8 presents an example of dynamic capacity optimization. The graph illustrates how resources within a single node/data-center are shared. The horizontal dashed orange line is the base capacity of the slices in the node, while the blue plot is the traffic. If traffic is below capacity, the network does not change resources. If the traffic in one slice exceeds its capacity, and other slices have spare resources, the algorithm tries to balance the resources. In our model this is realized by a simple change in the $C_{n,s}$ values, in reality it would require complex VMs migration, resource orchestration, etc.

Vice versa, Fig. 9 shows an example of dynamic traffic optimization. The graph illustrates how traffic between nodes of the same layer can be balanced (horizontal offloading). We do not report a graph showing traffic balancing between nodes of different layers (vertical offloading) as it would be almost the same. The orange horizontal line is the total capacity of the node ($C = \sum_s C_s$), the blue line is the total traffic. If the traffic is over the total capacity, it is not possible to apply resource balancing within the node as all the resources of that node are utilized. However, if there is *another* node with spare capacity, part of the traffic in excess could be re-routed to that node.

Both the above examples are applied to nodes in the *far edge* layer, but the same optimization is applied also to *near edge* of *core* nodes.

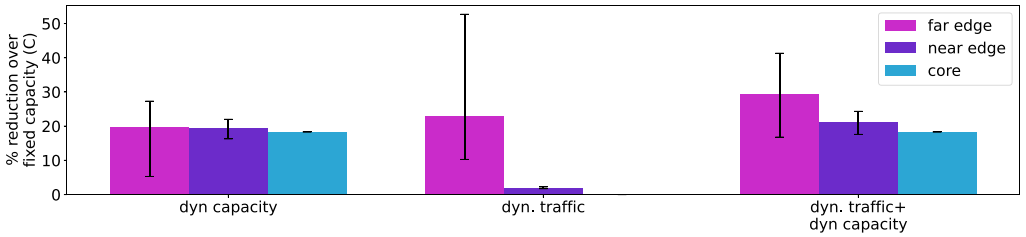

Fig. 10. Optimization results. The figure shows percentage reduction in traffic over capacity w.r.t the case of fixed capacity (C) set to handle 80% of hourly traffic. The first group of bars refers to dynamic capacity optimization (dyn. capacity). The second group of bars refers to dynamic traffic optimization (dyn. traffic). The third group of bars refers to a combination of the previous two optimizations. In each group, bars refer to far edge, near edge, and core layers. Combined results show that optimization achieves 29.5%, 21.1%, 18.3% percentage reduction in traffic over capacity for far edge, near edge, and core layers respectively. Error bars are the daily minimum and maximum over the experiment period.

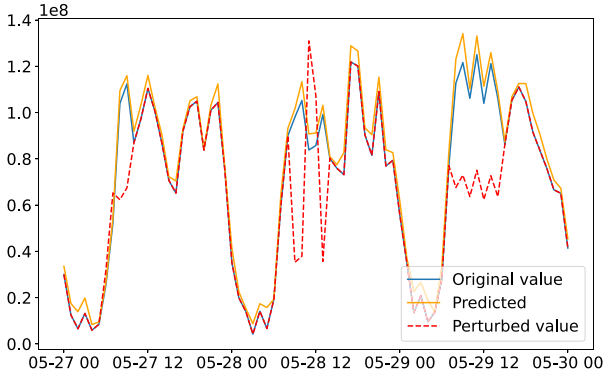

Fig. 11. Unreliable event information example (cluster 2, May 27–30). The dashed red line highlights corrupted values with CANC 30%, FP 20%, SHIFT 2 h.

Fig. 10 shows the result that can be achieved with such optimizations. In particular, we measure the percentage reduction in traffic-over-capacity achieved by the algorithm over the case of keeping fixed capacities and the original allocated traffic. Specifically, for each node, being TOC the Traffic Over Capacity in the static case and $OptTOC$ the Optimized Traffic Over Capacity, we compute $(TOC - OptTOC)/TOC * 100$. We then compute the average percentage reduction over all the nodes.

Results show percentage reduction using only dynamic capacity optimization, using only dynamic traffic optimization, and their combination.

Specifically, we observed a traffic reduction of 19.81% in far edge nodes, 19.44% in near edge nodes, and 18.31% in the core node, on average (see first group of bars). The second group of bars (dynamic traffic optimization only) achieves 23.06% traffic reduction over fixed capacity in far edge nodes, and almost zero (2.02% and 0.0%) in near edge ad core nodes. This is rather obvious as there are a lot of far edge node with which to share traffic, while there are very few near edge nodes (four) and only one core node. The third group of bars show the final combined optimization for the different layers: 29.45%, 21.06%, 18.31% traffic reduction over fixed capacity in far edge, near edge, core nodes respectively. Error bars represent the daily minimum and maximum of percentage reduction in traffic over fixed capacity the experiment period. It is possible to see that far edge nodes exhibit a greater variability than upper nodes because of the greater variability in traffic patterns.

For comparison, it is worth noting that standard forecasting algorithms (e.g., LSTM with RMSE loss function) tend to underestimate peak traffic. We conducted experiments with RMSE and verified a performance degradation of almost 40% compared to our proposed method. By contrast, our customized forecasting approach, with a slight tendency to overestimation, achieves better and more stable results across all three network levels.

5.4.1. Robustness to unreliable event information

As an additional experiment, we assessed the robustness of our framework against unreliable event information across all five clusters, fixing C at 80% for consistency with the baseline case shown in Fig. 10. We generated 27 scenarios ($3 \times 3 \times 3$) by perturbing event data along three axes: (i) cancellations (CANC), where 10%, 20%, and 30% of events were randomly suppressed, reducing the associated peaks by 40%; (ii) false positives (FP), where 5%, 10%, and 20% of non-event periods were artificially inflated by +25%; and (iii) temporal shifts (SHIFT), where about 20% of events were displaced by ±0, 1, or 2 h.

We then compared the perturbed forecasts against the baseline, quantifying both forecasting accuracy and operational impact.

Fig. 11 shows an example (cluster 2, May 27–30) where the perturbed value series for scenario 27 (CANC 30%, FP 20%, SHIFT 2 h) deviates from the original. In this case, the forecast RMSE increased from 5.72×10^6 to 1.29×10^7 (i.e., +125%), while the

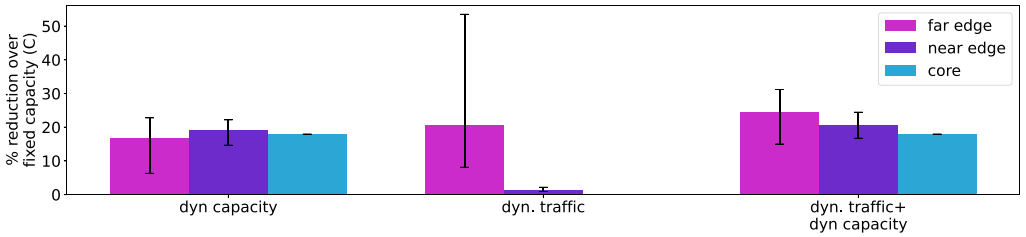

Fig. 12. Robustness analysis under unreliable event information for $C = 80\%$ and scenario 27 (CANC 30%, FP 20%, SHIFT 2 h). Compared to the baseline (Fig. 10), traffic reduction decreases to 24.4%, 20.6%, and 17.9% for far, near, and core nodes, respectively.

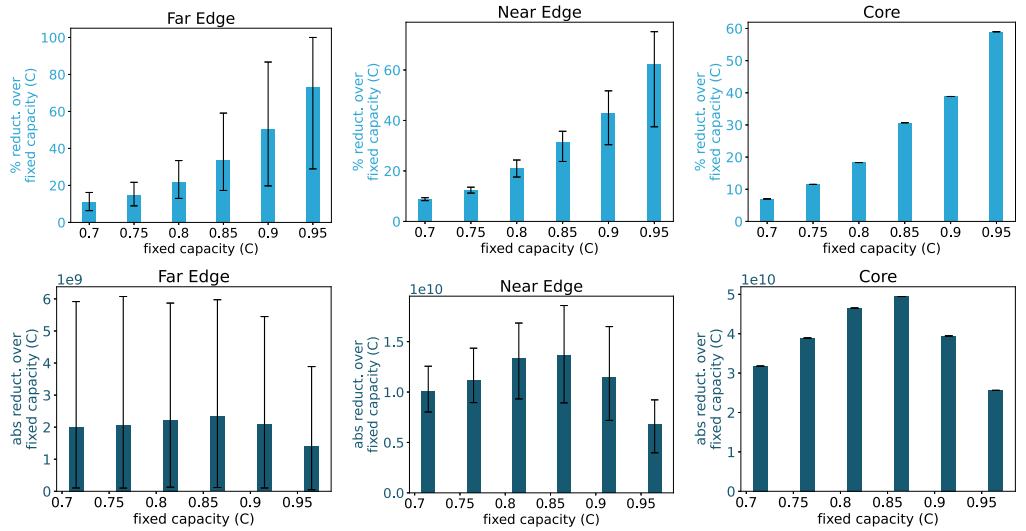

Fig. 13. Optimization results. The figure shows reduction in traffic over capacity w.r.t the case of fixed capacity (C) for different values of C. C is set to handle 70%, 75%, 80%, 85%, 90%, 95% of the hourly traffic (x-axis). The three graphs show results for far edge, near edge, and core layers. Results are obtained combining dynamic capacity and dynamic traffic optimization. **Top row.** Percentage reduction. **Bottom row** Absolute reduction.

naïve random-walk RMSE moved from 1.83×10^7 to 1.71×10^7 (-6.7%). On average across all clusters, the forecast RMSE increases markedly under corrupted event information, whereas the naïve random-walk RMSE remains comparatively stable, with variations up to 7%. This indicates that corrupted event information disproportionately penalizes the intelligent model, which strongly relies on event-based features, while the naïve baseline is less affected precisely because it does not exploit them.

Fig. 12 reports the corresponding optimization outcomes under scenario 26, to be compared with the baseline results in Fig. 10: reductions fall from 29.5%, 21.1%, and 18.3% (far, near, core) to 24.4%, 20.6%, and 17.9%, respectively.

Overall, the findings highlight that our framework is sensitive to unreliable event data; however, even under the most adverse scenario tested (30% cancellations, 20% false positives, and 20% of 2 h shifted events), the system continues to deliver meaningful forecasting accuracy and significant optimization gains, confirming its robustness to imperfect real-world information.

5.4.2. Impact of base capacity C on optimization results

The above results depend also on the base capacity C being adopted. According to the model depicted in Fig. 4, for example, if the base capacity is set to handle 100% of the traffic, there would be no need of our optimization at all.

Fig. 13 shows results of traffic reduction over fixed capacity, in the case of combined dynamic capacity and traffic, for different values of C. Fig. 13-top row shows percentage reduction in traffic, while 13-bottom row shows the absolute reduction. Having both this information is useful as for example (looking at the far edge plot) the system achieves about 70% traffic reduction over fixed capacity in the case of fixed C handling 95% of the cases. The total amount of traffic being saved is however small as the base C handle most of the traffic anyway. Also in this case, error bars represent the daily minimum and maximum of percentage/absolute reduction in traffic over fixed capacity the experiment period.

Finally, Fig. 14 presents the same results in the case of perfect prediction of future traffic which underlines the potential of the proposed optimization mechanisms. Results indicate a huge percentage reductions over the fixed case (e.g., about 60% reduction over the case of fixed capacity, when C is set to handle 90% of the traffic). On the other hand, it highlights the importance of

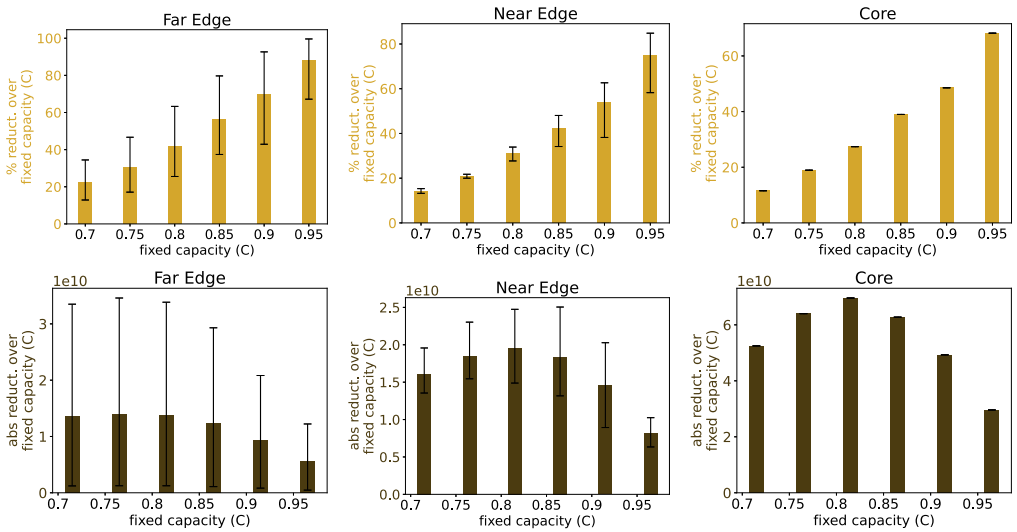

Fig. 14. Ideal case. This is the same graph as in Fig. 13 assuming traffic forecasting is perfect. **Top row.** Percentage reduction. **Bottom row** Absolute reduction.

traffic forecasting accuracy. Overall, these experiments demonstrate that the proposed approach is effective and could contribute to network management and optimization.

6. Related work

The flexibility introduced by the virtualization of 5G networks attracted a number of works trying to understand trade-offs in network design and management to improve network performance.

The problem of resource allocation of virtualized network functions within telco-cloud environments is the focus of a number of research [20–22]. Our work contributes to this line of the research exploring algorithms for resource and traffic re-balancing in the modeled scenario. Similarly, along these lines a number of works deal with the management and efficient deployment of network slices allowing a mobile network operator to efficiently allocate the underlying layer resources [23].

The seminal works [5,6] present a framework to understand and optimize network slicing in 5G networks. From a methodology view point, these works present an effective framework on how to model the mobile network. Our work has been largely influenced by their approach and we basically adopt the same model for the network. From the results' viewpoint, these works focus on the analysis of the use of dedicated vs. shared resources in mobile network slices. Our proposed approach of balancing resources between the network slices can be seen as an extension and complement to their framework.

The works in [4,24–26] present an optimization approach to assign network Access Points (APs) to MEC and edge servers. We applied a similar approach in the presented re-routing and load-balancing strategies. The main difference is that in our network model, an AP (or an edge server) can share part of its load with multiple edge servers above, while in these works each AP is dynamically connected with a single MEC instance (at least for a given network slice [25]).

The problem of resource allocation and traffic shaping for mobile network optimization has been largely studied with the use of optimization and game theory approaches [27–31]. In general, these works try to minimize a cost function associated to resource allocated to slices and network saturation under a number of constraint related to traffic and network capacity. Instead, our optimization algorithm combined with traffic forecasting proceeds with simplified assumption on the basis of the available data. Recently, a large number of proposals address this optimization problem using reinforcement learning (RL) and distributed reinforcement learning (DLR) techniques [32–35]. In our approach, we assume network's resource and traffic allocation can be dynamically reallocated around a nominal capacity C_s associated to each network slice. Therefore, RL, that is useful when a sequence of actions has to be inferred, is not suitable.

A large number of works in [36–40] deal with the problem of traffic forecasting in mobile networks. Most of the works deal with the forecasting of the overall network trend without focusing on specific areas covered by near edge or far edge nodes. On the other hand, other studies such as [15–17] delve into crowd prediction in various locales without addressing the network forecasting problem. The research by [15] utilized social media images to extract crowd size information during urban events, aiding in crowd management endeavors. Furthermore, the study by [16] demonstrated the utility of public social media data in tackling event detection and crowd level prediction in urban settings. Lastly, [17] examined the use of Twitter data analysis to estimate crowd sizes, underlining the significant potential of social media in contemporary society to support various research efforts, including crowd size estimation. Examining from the perspective of predictive Machine Learning (ML) techniques, there are alternative approaches, such as those presented by [41], which propose techniques divergent from neural networks for forecasting, claiming superior performance. However, the numerous assumptions made within these methods (e.g., user terminal traffic data can be profiled with reasonable

accuracy, user mobility and numbers in an area can be predicted with reasonable accuracy, etc.) render the proposed techniques less applicable to real-world scenarios. This incompatibility is further exacerbated by contemporary privacy concerns and data issues, which diverge significantly from the conditions under which the data utilized in [41] was collected and analyzed.

Our work, relying on additional information about events taking place, is able to forecast more accurately spikes associated with serving events. The works in [13,42] combine the forecasting approach with the resource optimization approach in a coherent model. We will investigate the application of similar models in our future work.

7. Conclusion

In this work we presented a modeling and optimization framework for resource and traffic sharing among virtual network servers and slices. Results show that dynamic reallocation of resources among slices improves performance by 19.19%, dynamic load balancing (traffic shaping) between nodes improves performance by 8.36%. Overall, the proposed data optimization allow to improve performance by 22.94%. The principal constraint of this study lies on the fact that, although we based our research on a scenario analogous to an actual deployment, we were compelled to make several assumptions regarding the data-related network architecture and its operations to compensate for the lack of certain information. Indeed, while our work presents a full-stack solution encompassing multiple modeling steps and optimizations, each individual component – such as network modeling, link capacity, latency requirements, service clustering, demand forecasting, and optimization – can be further refined.

In our future work, we aim to optimize these individual steps, drawing more detailed comparisons with the state of the art to enhance their effectiveness. In particular, an area of research that we plan to address is the dynamic offloading and caching of services and contents among the mobile network layers with a focus on the MEC layer [43]. While, the dataset analyzed in this work does not allow to carry on such an analysis, we plan to collect more detailed data for this kind of applications.

A crucial aspect of our future research will be to gain a deeper understanding of the underlying network structure and the available reconfiguration mechanisms. This will enable us to more firmly anchor our analysis in concrete scenarios thereby strengthening the practical applicability of our findings.

Finally, we believe that the main strength of this work lies in presenting a comprehensive solution that integrates multiple modeling and optimization steps. This holistic approach addresses various challenges in network analysis, offering valuable insights for further research and practical applications.

CRediT authorship contribution statement

Marcello Pietri: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing. **Natalia Selini Hadjidimitriou:** Data curation, Formal analysis, Validation, Visualization, Writing – original draft, Writing – review & editing. **Marco Mamei:** Conceptualization, Data curation, Formal analysis, Funding acquisition, Investigation, Methodology, Project administration, Resources, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing, Software. **Marco Picone:** Methodology, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing. **Enrico Rossini:** Conceptualization, Data curation, Writing – original draft, Writing – review & editing, Visualization. **Edoardo Maria Sanna:** Conceptualization, Data curation, Formal analysis, Methodology, Resources, Software, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing, Investigation, Funding acquisition. **Jovanka Adzic:** Conceptualization, Data curation, Formal analysis, Funding acquisition, Investigation, Methodology, Project administration, Resources, Software, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing. **Andrea Buldorini:** Conceptualization, Data curation, Formal analysis, Funding acquisition, Investigation, Methodology, Project administration, Resources, Software, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing.

Acknowledgments

This research was supported by the IPCEI – Cloud Infrastructure and Services project. The completion of this study, qualifying it as a spillover activity, required an essential self-funded investment by TIM beyond the original IPCEI financial scope.

This work has also received funding from the Chips Joint Undertaking (Chips JU) under the Ecomobility project, co-funded by the Horizon Europe Framework Programme and National Authorities (101112306).

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Research Link Provided

NetMob 2023 Data Challenge (Original data) (NetMob 2023 Data Challenge)

References

[1] ITU, Minimum requirements related to technical performance for IMT-2020 radio interface(s), 2017, URL https://www.itu.int/pub/R-REP-M.2410-2017.
[2] N. Zincir-Heywood, R. Birke, E. Bou-Harb, G. Casale, K. El-Khatib, T. Inoue, N. Kumar, H. Lutfiyya, D. Puthal, A. Shami, N. Stakhanova, F. Zulkernine, Special section on machine learning and artificial intelligence for managing networks, systems, and services, IEEE Trans. Netw. Serv. Manag. 20 (2) (2023) 882–889.
[3] O.E. Martínez-Durive, S. Mishra, C. Ziemlicki, S. Rubrichi, Z. Smoreda, M. Fiore, The NetMob23 dataset: A high-resolution multi-region service-level mobile data traffic cartography, 2023, arXiv:2305.06933.
[4] A. Ceselli, M. Fiore, A. Furno, M. Premoli, S. Secci, R. Stanica, Prescriptive analytics for MEC orchestration, in: 2018 IFIP Networking Conference (IFIP Networking) and Workshops, 2018.
[5] C. Marquez, M. Gramaglia, M. Fiore, A. Banchs, X. Costa-Perez, How should I slice my network? A Multi-Service empirical evaluation of resource sharing efficiency, in: Proceedings of the 24th Annual International Conference on Mobile Computing and Networking, MobiCom '18, New Delhi, India, 2018.
[6] C. Marquez, M. Gramaglia, M. Fiore, A. Banchs, X. Costa-Pérez, Resource sharing efficiency in network slicing, IEEE Trans. Netw. Serv. Manag. 16 (3) (2019) 909–923.
[7] P. Mach, Z. Becvar, Mobile edge computing: A survey on architecture and computation offloading, IEEE Commun. Surv. Tutor. 19 (3) (2017) 1628–1656, arXiv:1702.05309.
[8] S. Hochreiter, J. Schmidhuber, Long Short-Term memory, Neural Comput. 9 (8) (1997) 1735–1780.
[9] E.A. Mazied, D.S. Nikolopoulos, Y. Hanafy, S.F. Midkiff, Auto-scaling edge cloud for network slicing, Front. High Perform. Comput. Volume 1 - 2023 (2023).
[10] R.P. de la Loire sous Licence Ouverte v2.0 (Etalab), Observatoire 2G, 3G, 4G, 5G, 2023-06-23, URL https://data.anfr.fr/anfr/visualisation/table/?id=dd11fac6-4531-4a27-9c8c-a3a9e4ec2107.
[11] M.I. Malinen, P. Fränti, Balanced k-means for clustering, in: Structural, Syntactic, and Statistical Pattern Recognition: Joint IAPR International Workshop, S+ SSPR 2014, Joensuu, Finland, August 20-22, 2014. Proceedings, Springer, 2014, pp. 32–41.
[12] Z. Min, S. Gokhale, S. Shekhar, C. Mahmoudi, Z. Kang, Y. Barve, A. Gokhale, Enhancing 5G network slicing for IoT traffic with a novel clustering framework, Pervasive Mob. Comput. J. 104 (2024).
[13] D. Bega, M. Gramaglia, M. Fiore, A. Banchs, X. Costa-Perez, DeepCog: Cognitive network management in sliced 5G networks with deep learning, in: IEEE Conference on Computer Communications - INFOCOM, 2019, pp. 280–288.
[14] A.E. Beaton, J.W. Tukey, The fitting of power series, meaning polynomials, illustrated on Band-Spectroscopic data, in: Technomerics, Vol. 16 No. 2, Pp. 147-185, American Society for Quality, 1974, Introduces Tukey's bisquare (redescending) loss.
[15] A. Borthakur, A. Bhowmick, S.M. Hazarika, Crowd size estimation from social media images, in: 2019 International Conference on Computer, Electrical & Communication Engineering, IEEE, 2019, pp. 1–6.
[16] Y. Al Hammadi, I. Kamel, Evaluating publicly available data for crowd monitoring and event detection, in: IEEE/ACS International Conference on Computer Systems and Applications, 2017, pp. 764–769.
[17] A. Botta, W. De Donato, V. Persico, A. Pescapé, Integration of cloud computing and internet of things: A survey, Future Gener. Comput. Syst. 56 (2015) 684–700.
[18] T. Akiba, S. Sano, T. Yanase, T. Ohta, M. Koyama, Optuna: A next-generation hyperparameter optimization framework, in: ACM International Conference on Knowledge Discovery & Data Mining, New York, NY, USA, 2019.
[19] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, 2017, arXiv:1412.6980.
[20] A. Laghrissi, T. Taleb, A survey on the placement of virtual resources and virtual network functions, IEEE Commun. Surv. Tutor. 21 (2) (2018) 1409–1434.
[21] S. Clayman, E. Maini, A. Galis, A. Manzalini, N. Mazzocca, The dynamic placement of virtual network functions, in: 2014 IEEE Network Operations and Management Symposium, NOMS, 2014, pp. 1–9.
[22] R. Cohen, L. Lewin-Eytan, J.S. Naor, D. Raz, Near optimal placement of virtual network functions, in: 2015 IEEE Conference on Computer Communications, INFOCOM, 2015, pp. 1346–1354.
[23] R.A. Addad, M. Bagaa, T. Taleb, D.L.C. Dutra, H. Flinck, Optimization model for cross-domain network slices in 5G networks, IEEE Trans. Mob. Comput. 19 (5) (2019) 1156–1169.
[24] A. Ceselli, M. Fiore, M. Premoli, S. Secci, Optimized assignment patterns in mobile edge cloud networks, Comput. Oper. Res. 106 (2019) 246–259.
[25] C. Quadri, M. Premoli, A. Ceselli, S. Gaito, G.P. Rossi, Optimal assignment plan in sliced backhaul networks, IEEE Access 8 (2020) 68983–69002.
[26] B. Li, P. Hou, H. Wu, F. Hou, Optimal edge server deployment and allocation strategy in 5G ultra-dense networking environments, Pervasive Mob. Comput. J. 72 (2021).
[27] B. Jaumard, Q.H. Duong, A nested decomposition model for reliable NFV 5G network slicing, IEEE Trans. Netw. Serv. Manag. 20 (3) (2023) 2186–2200.
[28] A.F. Ocampo, M.-R. Fida, J.F. Botero, A. Elmokashfi, H. Bryhni, Opportunistic CPU sharing in mobile edge computing deploying the Cloud-RAN, IEEE Trans. Netw. Serv. Manag. 20 (3) (2023) 2201–2217.
[29] A. Awad Abdellatif, A. Abo-Eleneen, A. Mohamed, A. Erbad, N.V. Navkar, M. Guizani, Intelligent-Slicing: An AI-Assisted network slicing framework for 5G-and-Beyond networks, IEEE Trans. Netw. Serv. Manag. 20 (2) (2023) 1024–1039.
[30] M. Karbalaee Motalleb, V. Shah-Mansouri, S. Parsaeefard, O.L. Alcaraz López, Resource allocation in an open RAN system using network slicing, IEEE Trans. Netw. Serv. Manag. 20 (1) (2023) 471–485.
[31] C.C. González, E.F. Pupo, L. Atzori, M. Murroni, Dynamic radio access selection and slice allocation for differentiated traffic management on future mobile networks, IEEE Trans. Netw. Serv. Manag. 19 (3) (2022) 1965–1981.
[32] Y. Chiang, C.-H. Hsu, G.-H. Chen, H.-Y. Wei, Deep Q-Learning-Based dynamic network slicing and task offloading in edge network, IEEE Trans. Netw. Serv. Manag. 20 (1) (2023) 369–384.
[33] H. Li, K.D.R. Assis, S. Yan, D. Simeonidou, DRL-Based Long-Term resource planning for task offloading policies in multiserver edge computing networks, IEEE Trans. Netw. Serv. Manag. 19 (4) (2022) 4151–4164.
[34] J. Lou, Z. Tang, W. Jia, Energy-Efficient joint task assignment and migration in data centers: A deep reinforcement learning approach, IEEE Trans. Netw. Serv. Manag. 20 (2) (2023) 961–973.
[35] M. Sulaiman, A. Moayyedi, M. Ahmadi, M.A. Salahuddin, R. Boutaba, A. Saleh, Coordinated slicing and admission control using Multi-Agent deep reinforcement learning, IEEE Trans. Netw. Serv. Manag. 20 (2) (2023) 1110–1124.
[36] A. Furno, D. Naboulsi, R. Stanica, M. Fiore, Mobile demand profiling for cellular cognitive networking, IEEE Trans. Mob. Comput. 16 (3) (2017) 772–786.
[37] L. Lo Schiavo, M. Fiore, M. Gramaglia, A. Banchs, X. Costa-Perez, Forecasting for network management with joint statistical modelling and machine learning, in: 2022 IEEE 23rd International Symposium on a World of Wireless, Mobile and Multimedia Networks, 2022, pp. 60–69.
[38] W. Jiang, Cellular traffic prediction with machine learning: A survey, Expert Syst. Appl. 201 (2022) 117163.
[39] I. Alawe, A. Ksentini, Y. Hadjadj-Aoul, P. Bertin, Improving traffic forecasting for 5G core network scalability: A machine learning approach, IEEE Netw. 32 (6) (2018) 42–49.
[40] Z. Rao, Y. Xu, S. Pan, J. Guo, Y. Yan, Z. Wang, Cellular traffic prediction: A deep learning method considering dynamic nonlocal spatial correlation, Self-Attention, and correlation of spatiotemporal feature fusion, IEEE Trans. Netw. Serv. Manag. 20 (1) (2023) 426–440.

[41] K. Shiomoto, T. Otoshi, M. Murata, A network and computing resource management method based on population prediction in mobile networks, in: International Conference on Network of the Future, 2021, pp. 1–8.
[42] V. Sciancalepore, K. Samdanis, X. Costa-Perez, D. Bega, M. Gramaglia, A. Banchs, Mobile traffic forecasting for maximizing 5G network slicing resource utilization, in: IEEE Conference on Computer Communications - INFOCOM, 2017, pp. 1–9.
[43] A. Tian, B. Feng, H. Zhou, Y. Huang, K. Sood, S. Yu, H. Zhang, Efficient federated DRL-Based cooperative caching for mobile edge networks, IEEE Trans. Netw. Serv. Manag. 20 (1) (2023) 246–260.